Contract No. DAAH04-93-C-0019

OEIC Ultra
A New Approach to OEIC CAD

Author
R. Amantea

David Sarnoff Research Center
Princeton, NJ 08543-5300

October 17, 1994

Final Report
For the period June 1, 1993 through August 31, 1994

Prepared for
Army Research Office
Research Triangle Park, NC

The research reported herein has been sponsored by the U.S. Army Research Office under Contract DAAH04-93-C-0019. The views expressed herein do not necessarily reflect the position or policy of the U.S. Government.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	Oct 94	Final 1 Jun 93-31 Aug 94

4. TITLE AND SUBTITLE
OEIC Ultra. A New Approach to OEIC CAD

5. FUNDING NUMBERS
DAAH04-93-C-0019

6. AUTHOR(S)
R. Amantea

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
David Sarnoff Research Center, Inc.
Princeton, NJ 08543-5300

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
U.S. Army Research Office
P.O. Box 12211
Research Triangle Park, NC 27709-2211

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
ARO 31500.2-EL

11. SUPPLEMENTARY NOTES
The views, opinions and/or findings contained in this report are those of the author(s) and should not be construed as an official Department of the Army position, policy, or decision, unless so designated by other documentation.

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

Integrated optoelectronics will lead to advances in ultra-fast computing that go beyond the functionality that conventional electronics can reach through size reduction. Optoelectronic integrated circuit (OEIC) design is currently hampered by the lack of a unified computer-aided design (CAD) tool that aids in managing the complex data implicit in component, subsystem, and system-level modeling.

To address the advanced CAD requirements of ULTRA, the David Sarnoff Research Center (Sarnoff) has created a hierarchical OEIC design system built using a platform-independent simulation and visualization environment supported by commercial-off-the-shelf software. Sarnoff has demonstrated integration of existing modules by graphically "wiring" them up on-screen to represent the complete design problem. Sarnoff has demonstrated how integrated visualization tools can support the design process by allowing automatic exploration of the effects of parameters and structural features on lasers and OEIC devices while the system automatically manages data flow among modules.

(continued on reverse side)

14. SUBJECT TERMS	15. NUMBER OF PAGES
Optoelectronics, Integrated Circuits, Computer Aided Design, Computerized Simulation	103
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

This CAD environment will enable the ARPA ULTRA program to exploit optoelectronics at a faster pace than achievable by the stand-along design approaches in use today.

Table of Contents

List of Illustrations

List of Illustrations (Cont'd.)

Abbreviations

BPM:	Beam propagation method
CAD:	Computer-aided design
COTS:	Commercial-off-the-shelf
GUI:	Graphics user interface
MSI:	Medium scale integration
OEIC:	Optoelectronic integrated circuit
SVE:	Simulation visualization environment
VLSI:	Very large scale integration

Abstract

Integrated optoelectronics will lead to advances in ultra-fast computing that go beyond the functionality that conventional electronics can reach through size reduction. Optoelectronic integrated circuit (OEIC) design is currently hampered by the lack of a unified computer-aided design (CAD) tool that aids in managing the complex data implicit in component, subsystem, and system-level modeling.

To address the advanced CAD requirements of ULTRA, the David Sarnoff Research Center (Sarnoff) has created a hierarchical OEIC design system built using a platform-independent simulation and visualization environment (SVE) supported by commercial-off-the-shelf (COTS) software. Sarnoff has demonstrated integration of existing modules by graphically "wiring" them up on-screen to represent the complete design problem. Sarnoff has demonstrated how integrated visualization tools can support the design process by allowing automatic exploration of the effects of parameters and structural features on lasers and other OEIC devices while the system automatically manages data flow among modules.

This CAD environment will enable the ARPA ULTRA program to exploit optoelectronics at a faster pace than achievable by the stand-alone design approaches in use today, resulting in:

- Integrated design teams
- Shorter design cycles
- Reduced development risk
- Improved transfer to manufacture
- Integration of diverse development efforts
- Cost-effective designs
- Migration to CAM

1. Executive Summary

Optoelectronic components can potentially lead to advances in ultra-fast computing that go beyond the functionality that conventional electronics can reach through size reduction. Realization of the potential of optoelectronics to enable higher performance computational systems will require advanced CAD tools that integrate electrical, optical, and optoelectronic device analysis within a unified design environment. The complexity, and potentially conflicting requirements, of these different components causes the application of existing individual design tools to be very difficult. At present, there are no general-purpose CAD systems that can be applied to simulating and optimizing integrated optoelectronic components and subsystems. Today commercial field propagation programs either lack generality or are extremely difficult to use. Optoelectronic software developed at universities, although useful, is usually limited to special purposes and is generally dificult to use by anyone other than the developer. This work addresses these issues by utilizing a commercial visualization environment for a consistent user interface that allows easy import of new computational modules.

The benefits of new CAD tools for efficient and rapid progress in the ULTRA domain are borne out by the history of silicon VLSI. The transition from small scale integration to VLSI led to the development of a new way of thinking about the design process based on the hierarchical nature of large systems. This gave the designer the ability to use circuit modeling, chip layout, and other CAD tools. These tools allowed the full exploitation of technological progress that in turn resulted in the rapid transition from MSI to VLSI. Optoelectronics technology has reached the stage of maturity at which the development of appropriate CAD tools will provide similar benefits.

Sarnoff has applied the lessons learned in the development of CAD systems for VLSI to develop an OEIC CAD system that is optimized to simulate integrated guided wave structures. The CAD system is also optimized to visualize the resulting electromagnetic fields, carrier densities, and other related physical quantities.

2. Background

The ULTRA program requires sophisticated, flexible CAD tools incorporating electronics, optics, and optoelectronics to realize complex high-performance optoelectronic circuits for optical computing and interconnects. Sarnoff is using a hierarchical design approach embedded in a distributed-computing SVE that will provide the ability to design and analyze complex systems, subsystems, components, and new devices. In developing the OEIC design environment, we take advantage of the graphical programming environment and data visualization tools offered by COTS SVE software. Effort is concentrated toward developing an initial set of software modules for optoelectronic simulation by importing existing models and working out the details of optoelectronic data management within the SVE. This provides the basis for future work leading to a fully integrated optoelectronic CAD system that meets the evolving needs of ARPA's ULTRA program. Thus, we have developed software that is capable of modeling light propagation in a wide variety of geometrical configurations.

Conventional silicon-VLSI CAD handles only electronic signals. An analogous approach for optoelectronic components might treat optical and electronic signals as a function of time, such as pulse rise times, delays, current and voltages, etc., but neglect physical optical effects. Alternatively, a purely optical analysis might use electromagnetic techniques to determine eigenmodes, optical coupling, etc., which are functions of position in 3-dimensional (3D) space. Using current tools to design a complex OEIC requires a designer to apply individual electronic and optical models to each device and structure element. He or she has to manually organize and repetitively transfer data files and visualizations of the separate simulations and establish his or her personal interpretation of the complete circuit.

Addressing optoelectronic circuits and devices generally requires mixed-mode simulators, i.e., those capable of dealing simultaneously with systems of ordinary and partial differential equations for both electronic and optical properties. This approach, with its heavy computational load, is not practical for large systems. However, optoelectronic interactions in large systems can instead be modeled efficiently by a hierarchical SVE approach, as described in this report. With the use of the numerically efficient technique of *emulation*, in which the result of time-consuming computation is stored for later use, the capabilities of mixed-mode simulation can be retained.

In large systems of any kind, the need to manage complexity naturally leads to a hierarchical description. The overall function is broken down into subsystems, whose components are described in increasing detail at lower levels of the hierarchy. An example is shown in Fig. 1, which depicts portions of a multi-channel optoelectronic transmitter-receiver system represented by building blocks. We have subdivided the design into four levels that distinguish critical views of the system. The highest is Level **I**, a system level that describes optical communication system goals and specifications. Level **II** breaks the system down into more manageable subsystems that may be handled primarily with either electronic or optical analyses. Device design, modeling, and simulation takes place at Level **III**. Physical device simulators at Level **IV** calculate the internal physical behavior of individual devices. According to this view, the system consists of major building blocks, which in turn, consist of their constituent building blocks. In Fig. 1, the transmitter building block is made up of constituent distributed feedback (DFB) oscillator, amplifier, beam splitter, switch, and grating emitter blocks.

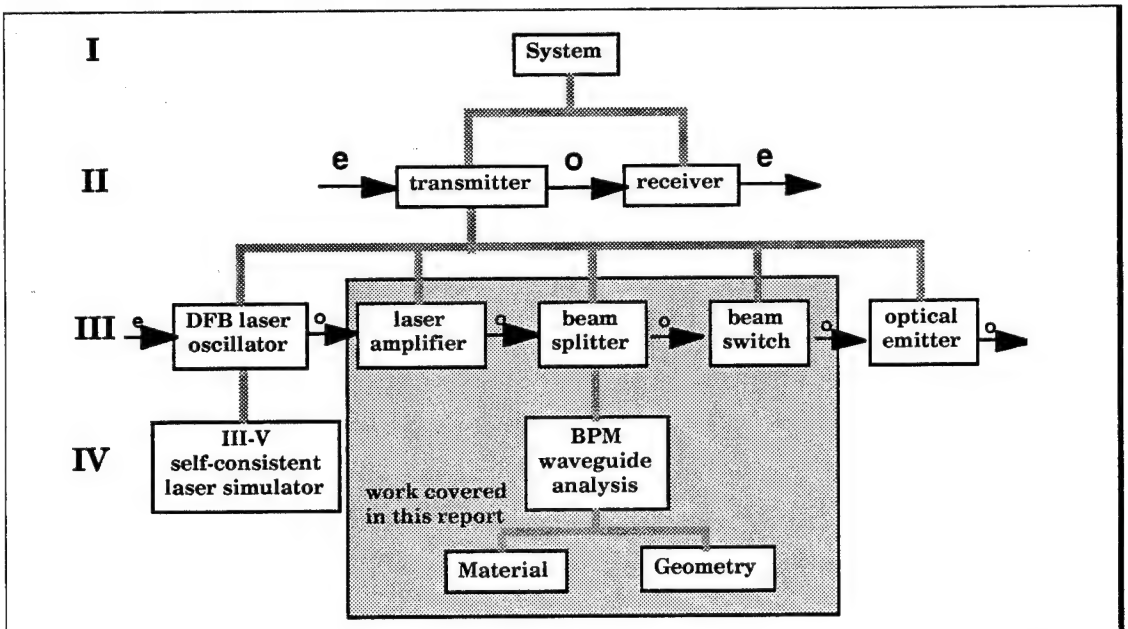


Figure 1. *A top-down analysis of multi-channel transmitter-receiver shows the different levels of analysis that need to be considered. Each block represents an actual element of the complete system. Higher level blocks are built by combining lower level blocks. The shaded area highlights the work reported here.*

A key advantage demonstrated by our methodology is the implementation of methods to deal with *data conversion* between computational modules. The importance of data conversion is that it allows the user to configure the simulation network with ease and it facilitates techniques that permit the simulation to run more quickly and efficiently. Furthermore, it makes incorporation of new computational modules much easier.

When a designer works at Level **I**, **II**, or **III**, he or she operates the CAD system to simulate a particular system, subsystem, or device. That process requires signal flows and model parameters generated by related software, which may represent lower level sub-systems, devices, or physical simulators. For example, working at Level **III** on the DFB laser oscillator we are simulating its behavior as part of an OEIC. It takes an electrical input signal and generates an optical output signal. To do this, we may address a Level **IV** physical device simulator to produce the operation parameters needed for the DFB laser-oscillator. On the other hand, to work at Level **II** on the transmitter, we address the Level **III** device models by connecting the DFB laser oscillator to the laser amplifier, optical switch array, etc.

The *signal flows* in the system being simulated are denoted by black arrows, labeled "e" where carried electronically, and "o" where carried optically. The critical components where device operation and behavior are hardest to model are where the information is converted from one type of flow to another, i.e., the optoelectronic components. These critical components define natural transitions for partitioning the system into subsystems that can be treated either as electronic or optical circuits. These components also define where greater computational effort is needed.

A powerful feature demonstrated by our approach is that data exchange between simulators and hierarchical levels, including any necessary numerical conversions (e.g., grid spacing), is a user-transparent process. Higher level models are constructed to depend on lower levels, with such *data dependencies* represented here by thick gray lines. For example, working at Level **II** on the transmitter, we need to find the input and output characteristics of the transmitter without all the intermediate results developed by the Level **III** device simulators. The necessary signal data and parameter data are transmitted upward automatically to the transmitter model.

As another illustration, the input to the DFB laser oscillator at Level **III** is an electronic signal. To generate an optical transverse field the designer of the device module drops to Level **IV**. There, he or she may apply a self-consistent III-V device simulator that generates the lateral (in-plane, perpendicular to direction of propagation) optical field. As discussed below, a physical device simulator that we will implement for

the amplifier simulation is a 2-dimensional (2D) unidirectional optical beam propagation model (BPM). The signal flow inputs required by this simulator are provided by the DFB laser oscillator module.

Since existing CAD tools are stand-alone, limited in applicability, and technology-dependent, the most immediate need for ULTRA is for integrated, broadly applicable, technology-independent tools that are directed toward analysis at Levels **III** and **IV**. In this program, we will first address the data flow along a single level of the hierarchy, e.g., Level **III**, because this will give us a useful CAD tool. We will structure the data to facilitate the vertical flow between hierarchical levels. Thus, an integrated CAD tool will evolve as the program progresses.

The resulting CAD system exhibits the following properties:

- An easy-to-use intuitive graphical user interface
- A large variety of data handling tools encompassing all of the necessary data types, including the multidimensional data
- A large variety of graphical viewing tools to portray data as curves, surfaces, or images
- The ability to import existing optoelectronic models
- Support for distributed processing that is transparent to the user
- Ability to run on a variety of workstation platforms, e.g., Sun Sparc Station 2, Macintosh, IBM PC, etc.

3. Task Performance

Task 1: Design of Optoelectronic CAD Environment

Objective

Design a hierarchical optoelectronic CAD environment using a COTS SVE software package.

Accomplishments

The desired COTS SVE software had to meet the following requirements:

- It must easily handle large data sets, e.g., several 500 by 500 two-dimensional arrays of extended precision floating point numbers.
- It must be capable of operating over a network.
- It must supply graphics tools such as line plots, contour plots, images, and surface plots.
- It must provide utilities to easily read and write large data files and to manipulate data.
- It must provide a GUI for specifying and manipulating data.

We have selected AVS[1] as the COTS SVE software. AVS is an application visualization system that provides a framework for the implementation of visual applications. It provides sophisticated computer graphics to gain understanding of complex problems that arise from large data sets. It integrates computation, analysis, and graphics. Not only does AVS meet the aforementioned requirements but it also provides:

- An iconic programming language for interconnecting modules where data flow between computational processes is configured by connecting icons
- Automatic data type checking between modules to prevent the wrong type of data being read by a module
- Predefined data types that meet the needs of multi-dimensional simulations such as vector fields over n-dimensional space

[1] AVS is a trademark of Advanced Visual Systems Inc., Waltham, MA.

The CAD system is partitioned into four groups of modules: materials, geometry, electromagnetic field processing, and data visualization tools as shown in Fig. 2. Each of these groups can be considered a level in a design hierarchy. Note that the groupings closely mirror the fabrication and test process. The formulation of a simulation follows a top-down methodology from materials through visualization. Data is accumulated at each level in the hierarchy and flows downward to the next level. Once specified, any parameter should not have to be specified again. This not only ensures against errors but it streamlines data input.

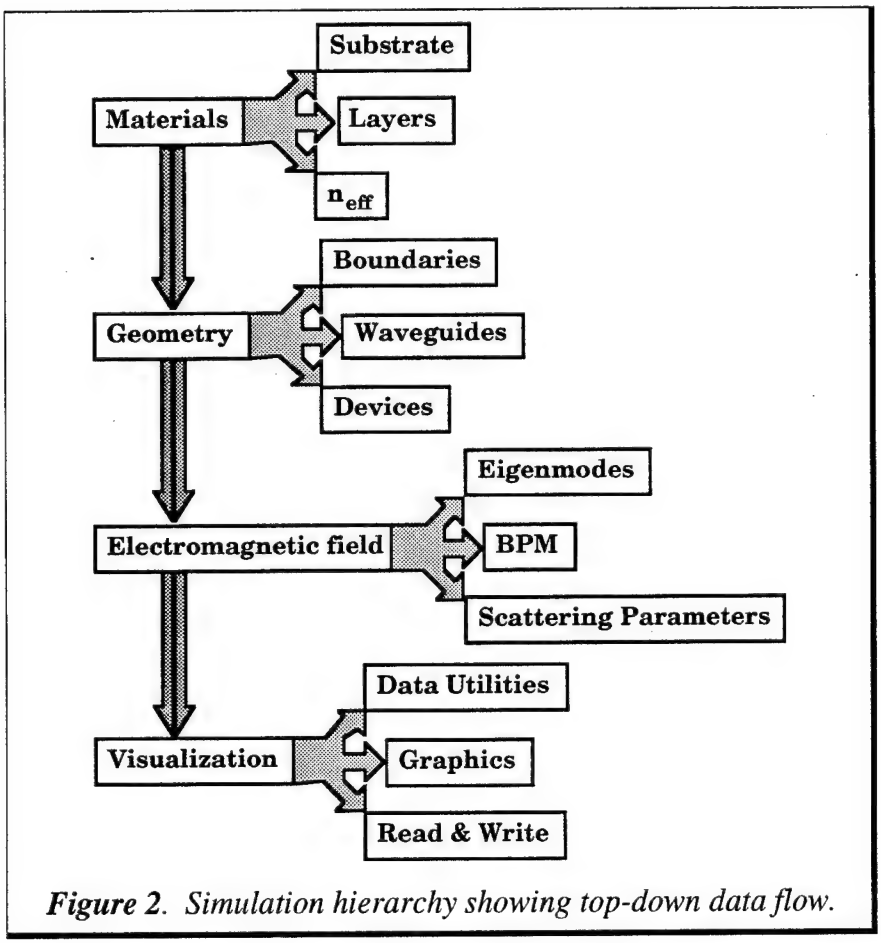


***Figure 2**. Simulation hierarchy showing top-down data flow.*

The material group of modules describe the process of specifying and designing the material layers in an OEIC based on the III-V compound semiconductors. These modules can be further divided into three types of modules: the substrate, material layers, and modules to convert this information into effective index. At the substrate level we must specify the basic properties of the material including wavelength. As each layer is

added we add to the specification layer properties such as composition and thickness. Finally, the composite material is summarized as an effective index by a 1-dimensional eigenmode finder for layered materials.

In a similar manner, the geometry group that describes two-dimensional rib or channel waveguides is also subdivided to deal with boundaries, waveguides, and devices. The boundaries specify the physical extent of simulations. For now we are dealing with two-dimensional regions where the lateral extent, i.e., the x direction, the longitudinal extent, i.e., the z direction, and the computational grid need to be specified. The transverse extent (y direction) is modeled by the effective index approximation. Straight and curved, uniform and nonuniform, passive and active waveguides also are specified. Finally, more complex geometries (such as y-splitters) can be defined to simplify problem specification.

Modules to compute electromagnetic field propagation in the OEIC were added to the system. A minimum useful set consists of a lateral eigenmode simulator to find waveguide modes and a flexible, comprehensive BPM for propagating the field through the structure. A mode summation module that computes complex lateral fields from the waveguide modes is necessary to provide inputs to the BPM. A mode projection module that allows one to calculate the modal spectrum of an output lateral field is necessary to develop generalized scattering parameters for subsequent use in optical circuit analysis.

Task 2: Development of Optoelectronic CAD Tools

Objective

On the basis of the system architecture defined in Task 1, Sarnoff will develop modules for simulating and designing an optoelectronic system.

Accomplishments

CAD components

Each problem to be solved in the OEIC CAD system is described by a network. A network is composed of modules, macro modules, wires, and panels. A module is an independent process that is started when valid data is presented to its input ports. Valid data appears at the output ports when the process completes. The ports are color coded according to the data type. In this report, ports and wires are shown in shades of gray.

A typical module is shown in Fig. 3. It consists of input ports, output ports, a panel box, and a title. Wires connect modules to one another and represent the data flow between them. A macro module is a collection of modules that have been encapsulated so that internal ports are hidden. Macromodules are distinguished by a blue box (dark gray in this report) on the right-hand side. They are opened by shift-clicking in the module rectangle.

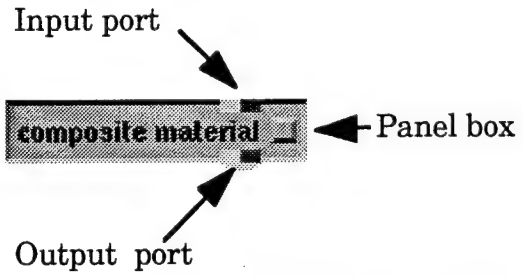


Figure 3. *A typical module as it appears in a simulation network.*

Panels are windows where the module parameters can be specified. They are opened by clicking in the panel box on the right-hand side of the module.

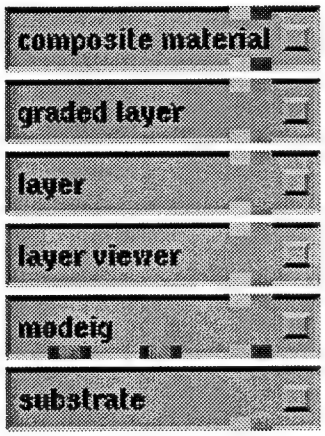


Figure 4. *Material modules are chosen from the module menu.*

Materials

There are six modules in the material group, as shown in Fig. 4. These modules are used to specify the transverse properties of the OEIC. Since the present system incorporates only two-dimensional analysis techniques, these modules are used to define the effective index of different regions on the wafer. Module function is summarized below and in the Appendix.

Module Name	Description
Composite material	Directly specifies the effective index of a region
Graded layer	Specifies the properties of a linearly graded layer
Layer	Specifies the layer properties
Layer viewer	Views the layer properties
Modeig	Computes the effective index of a series of layers on a substrate
Substrate	Specifies the substrate properties

The network shown in Fig. 5 computes the effective index of two layers over a substrate. In this example, the five modules completely specify all the material properties necessary for a simulation. Also shown in Fig. 5 is a macro module. Material specification begins with the substrate, whose panel is shown in Fig. 6. Here the material system and wavelength are chosen.

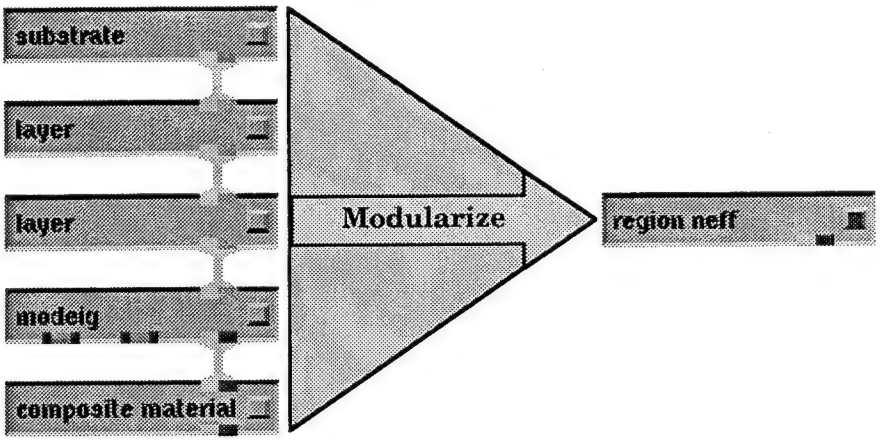


Figure 5. *A simple network is used to calculate the effective index of two layers on a substrate. It can be encapsulated into a macro module.*

The substrate is followed by one or more layers. A typical layer panel is shown in Fig. 7. After all layers have been specified the modeig module is connected to calculate the transverse eigenmode and the effective index. The modeig module uses a method simular to the transmission matrix method to satisfy electromagnetic boundary conditions between the layers. A coupling polynomial search procedure is used to find the eignemodes. The panel for modeig is shown in Fig. 8. Presently, all the capabilities of MODEIG are not used by the CAD system. Future work will bring the module performance up to its present stand-alone capability.

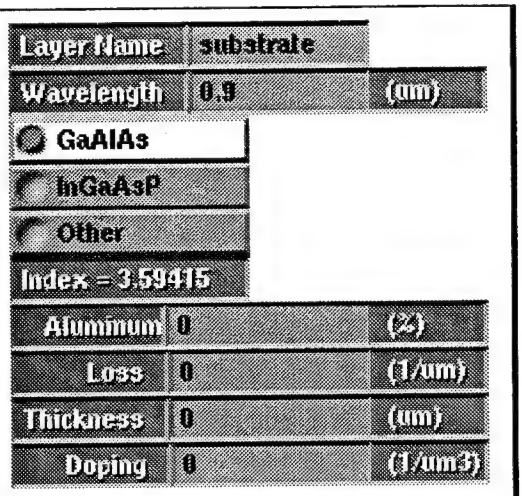


***Figure 6**. The substrate panel.*

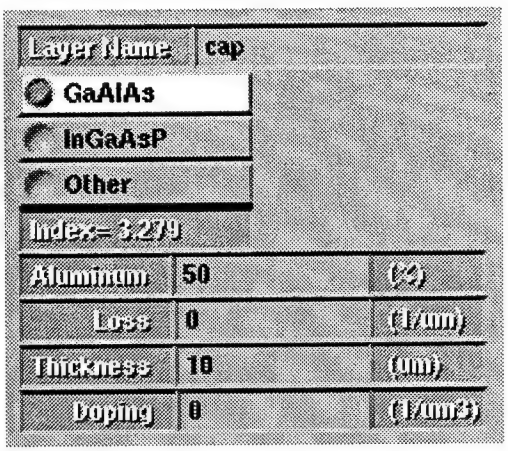


***Figure 7**. The layer panel.*

The final step in specifying the material is connecting the composite material module. The panel for the composite material module is shown in Fig. 9. This module translates the output of modeig into a format acceptable to the geometry modules. It also may be used in an emulation mode, to specify the effective index directly as an alternative to specifying the substrate and layers. Finally, it is used to specify all the remaining material parameters that are necessary for BPM simulation. The meanings of all the parameters are given in the Appendix.

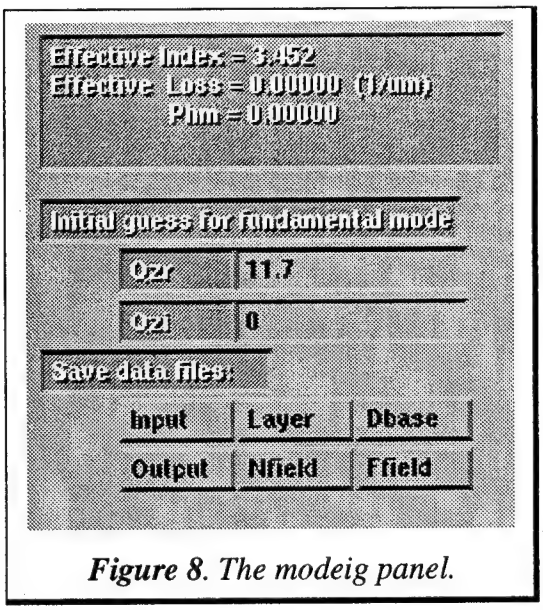


Figure 8. The modeig panel.

Any problem network can be split into multiple paths so that all the previously specified data can be shared by subsequent modules. This type of network is shown in Fig. 10. Here we see a substrate followed by two layers. After these two layers, the network splits into two and each material region is capped by a different layer. The results of these two regions are individually passed to a modeig module where the effective index is calculated.

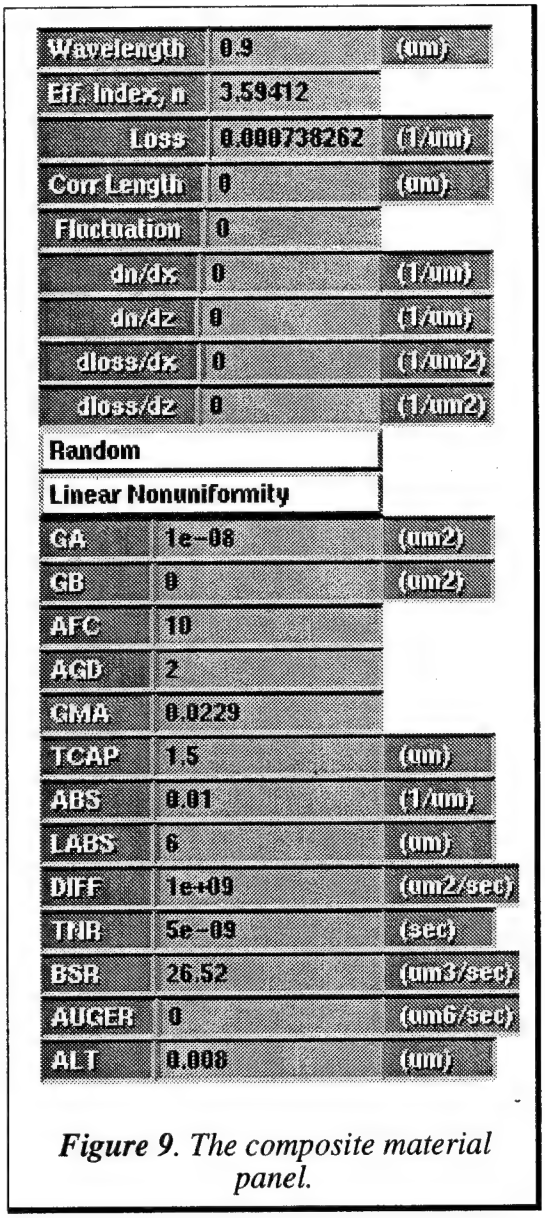


Figure 9. The composite material panel.

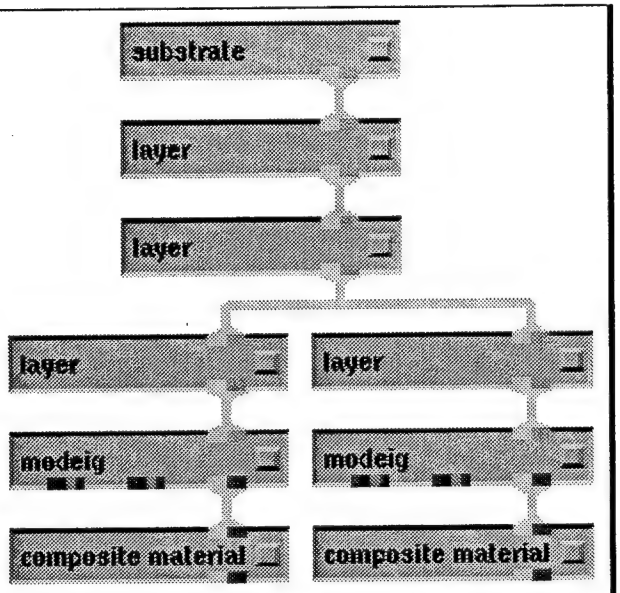


***Figure 10**. Two region specification.*

This procedure produces an effective index specification for two regions that differ only in the final layer. Consequently, data for the substrate and initial layers is shared and there is no possibility of introducing an incompatibility. Furthermore, if the initial layers are modified, the modification appears in both regions simultaneously without any interaction of the user.

Geometry

Figure 11 shows the geometry modules as they appear in the menu. A geometry network is composed of a domain module, region modules, and a capture geometry module. The domain module specifies the region over which the field propagation is solved. The region modules specify the properties of waveguides, structures, and devices. The capture geometry module collects all the geometric information.

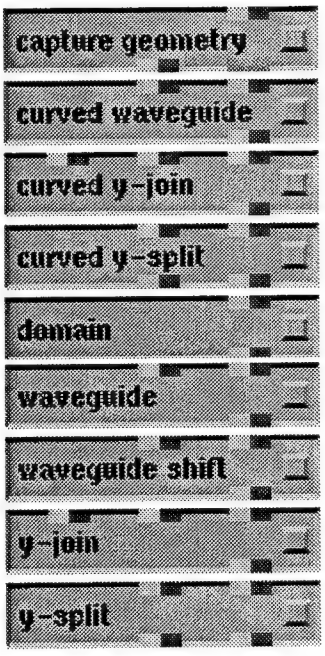


***Figure 11**. Geometry modules are chosen from the main module menu.*

The creation of geometry region modules involves a trade-off between individual module complexity and network complexity. For example, many geometries can be assembled by just using the simple waveguide module that provides for skew and flare. Although this limits the geometry to having straight edges, it is possible to approximate any complex geometry using this method.

However, to model curved regions, the number of modules increase and the ability to interpret the problem network is diminished. For this reason we have also included more complex geometries, including curved structures.

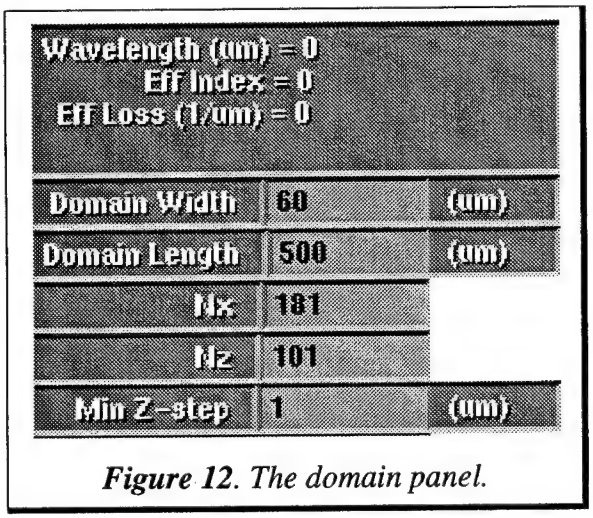


***Figure 12**. The domain panel.*

The domain panel is shown in Fig. 12. The domain is used to specify the simulation region extent, i.e., width and length. It also specifies the simulation grid, i.e., the minimum step along the direction of propagation, the number of lateral grid points, Nx, and the number of output field points in the longitudinal direction, Nz.

The waveguide module defines a ridge waveguide that is characterized by a uniform transverse effective index. The panel is shown in Fig. 13. The waveguide length in the z direction and initial width are specified in microns. The waveguide may be linearly tapered with an angular flare measured in degrees. The waveguide may also be skewed with respect to the z direction, and the skew is given in degrees. The geometry may be offset from its initial position in either the x or z direction. The geometry is shown in Fig. 14 with width = 3 μ, length = 500 μ, flare = 1°, and skew = 0.5°. Of course, when the flare and skew are zero then the resulting waveguide is uniform.

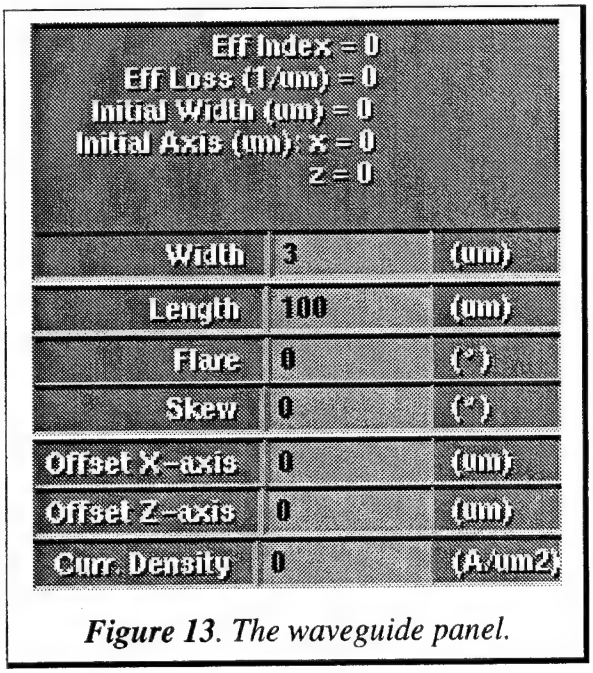


***Figure 13**. The waveguide panel.*

Geometry Modules	
Module name	**Description**
Capture geometry	Collects all geometry data and formats it for the bpm
Curved waveguide	Specifies a curved segment of waveguide
Curved y-join	Specifies a beam joiner composed of curved segments
Curved y-split	Specifies a beam splitter composed of curved segments
Domain	Specifies the region of interest and the numerical grid
Waveguide	Specifies a general waveguide
Waveguide shift	Specifies a pair of curved waveguides
Y-join	Specifies a beam joiner
Y-split	Specifies a beam splitter

If the waveguide is connected to and succeeds another region module then the width is initially set to the preceding region width and the width parameter becomes a change width parameter. The initial position of a region is set by the ending position of the preceding region. The offset parameters can then be used to move the region to an arbitrary position in the domain. This simplification speeds up the process of data specification.

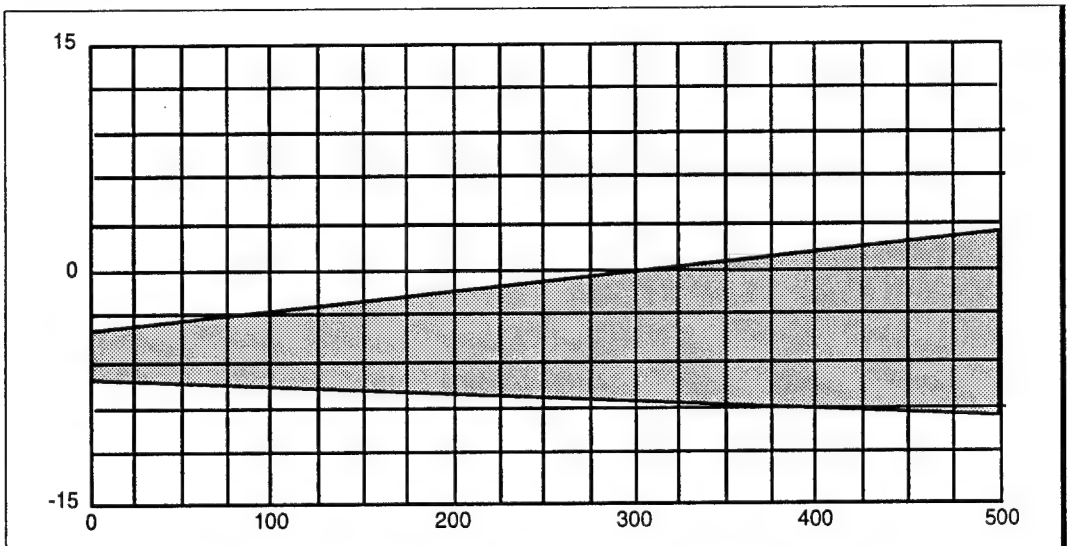


Figure 14. *Waveguide geometry with both flare and skew.*

Figure 15 shows the panel for a segment of curved waveguide. This primitive region is based upon a circular arc. The arc length, θ, radius of curvature, r, lateral shift, Δx, and length, l, are related by the equations

$r = l/\sin\theta$ and $\Delta x = \frac{l(1-\cos\theta)}{\sin\theta}$. The geometry is shown in Fig. 16 with width = 3 μ, length = 500 μ, and arc length = 1°. Note that it is possible to flip this geometry horizontally and vertically.

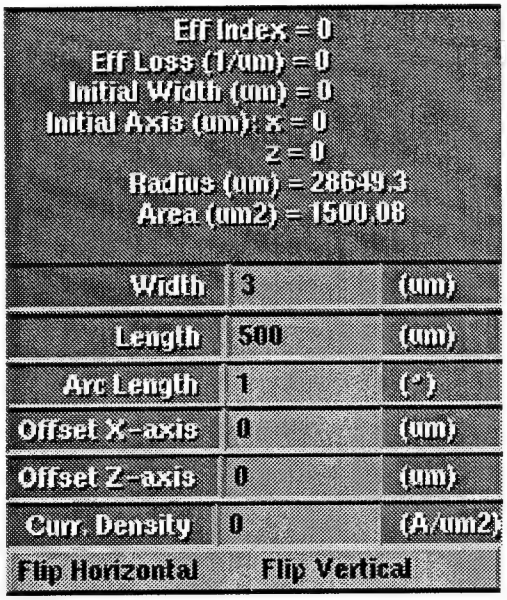


Figure 15. *Curved waveguide panel.*

We have found that primitive regions such as the curved waveguide are useful for building arbitrary geometries, however, most often all that is required is a smooth lateral shift.

The waveguide shift module, whose panel is shown in Fig. 17, meets this need by providing a pair of curved waveguides that smoothly guide the optical field. The device is based upon two circular arcs that are tangent to one another at the midpoint of the region. The same algebraic relations that applied to the curved waveguide apply here, however, here the total lateral offset is specified. This geometry is shown in Fig. 18.

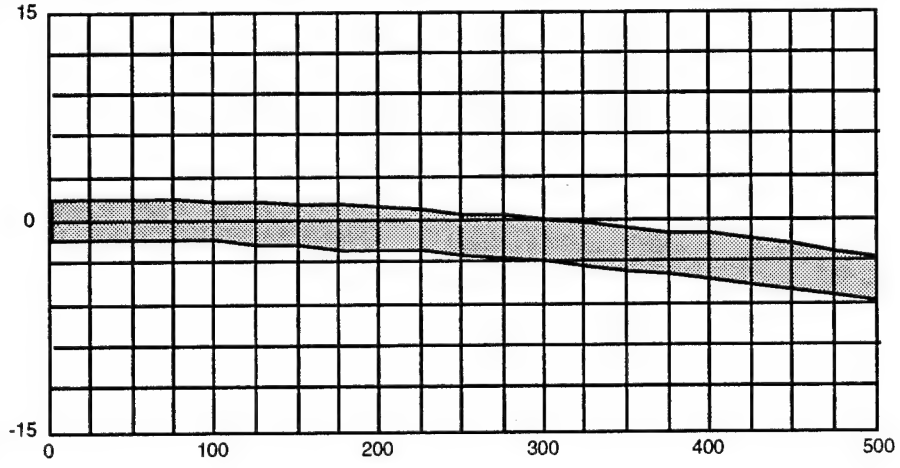


Figure 16. *Waveguide geometry of a curved waveguide.*

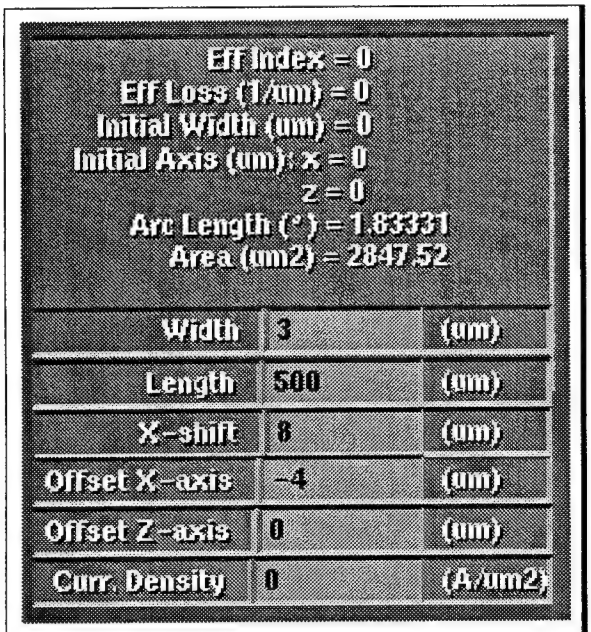


Figure 17. *The waveguide shift panel.*

Some examples of geometric configurations that can be assembled using these primitive geometric regions are shown in Figs. 21, 22, and 23. Y split and Y join devices are also available. These waveguide splitters can be specified as piece-wise linear or smoothly curved geometry. The panel for this region is shown in Fig. 19. Several additional parameters are defined so that imperfections can be modeled. The parameter *delta* specifies a difference in the width of the two branches and the parameter *delta split z* specifies the amount of the notch that is not etched as shown in the darkened region of the geometry shown in Fig. 20.

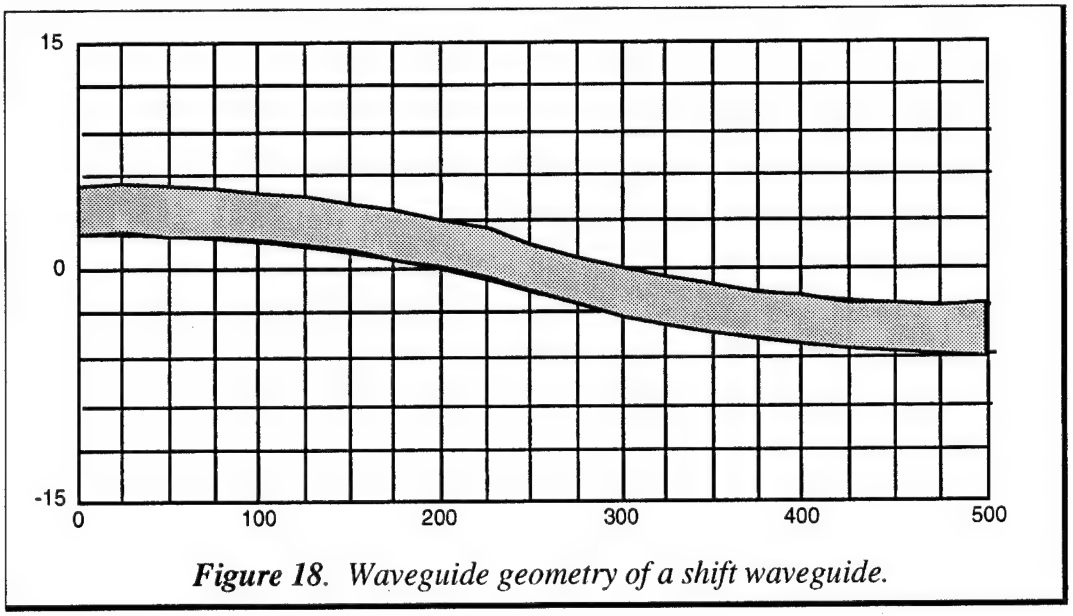


Figure 18. *Waveguide geometry of a shift waveguide.*

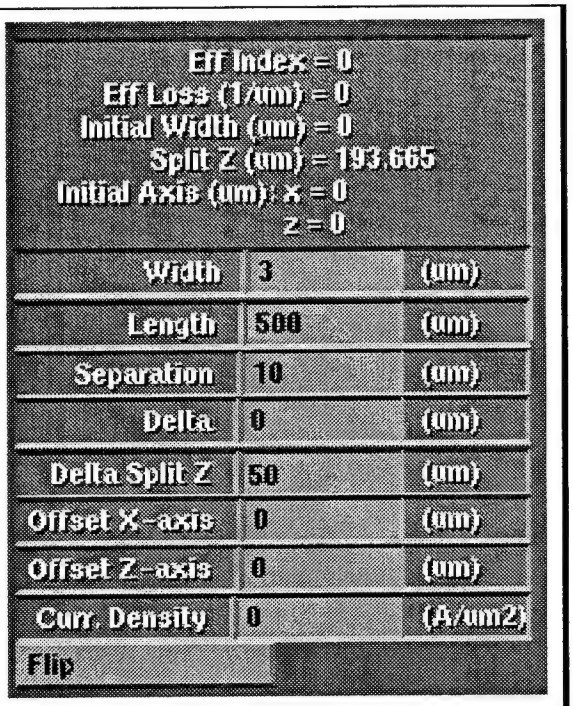


***Figure 19**. Y split module panel.*

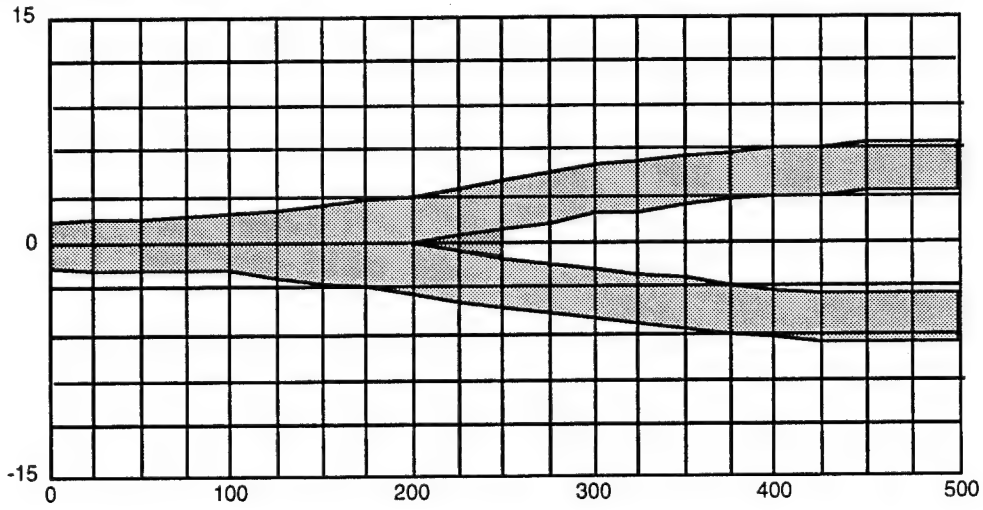


***Figure 20**. Waveguide geometry of a curved y split.*

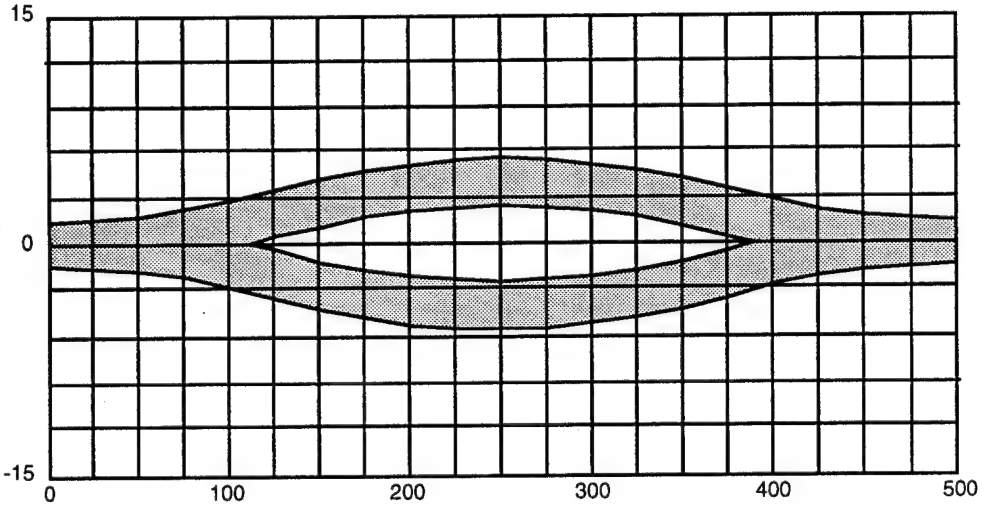


Figure 21. *Mach Zehnder interferometer.*

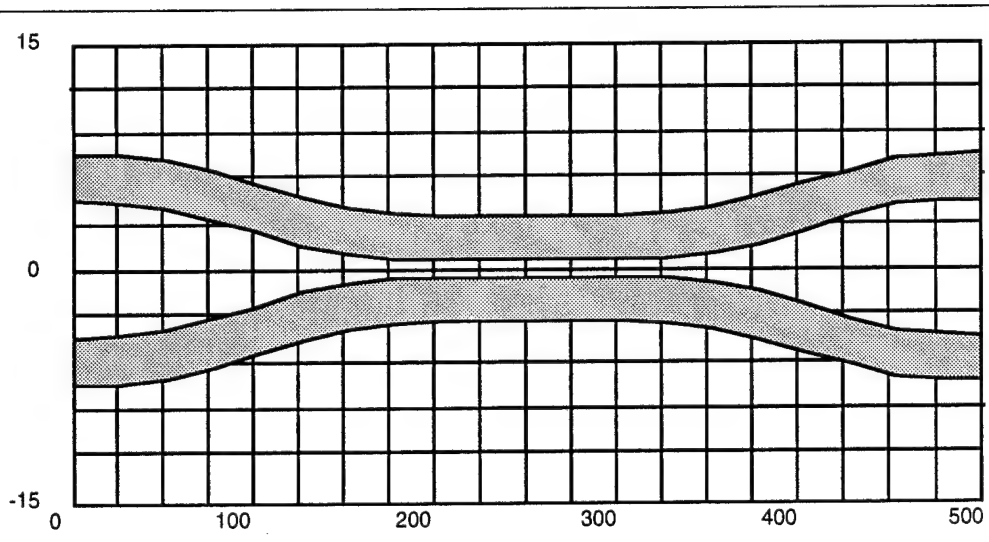


Figure 22. *Directional coupler.*

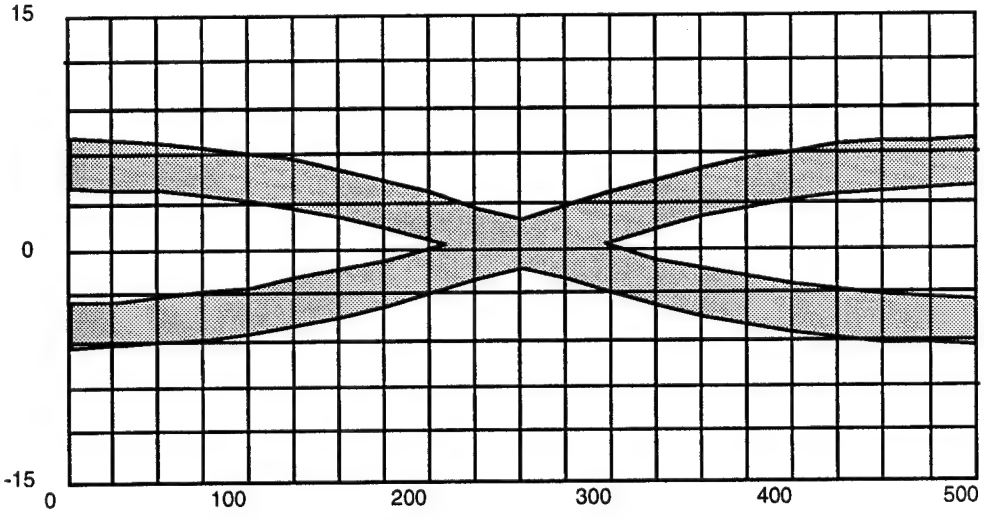


Figure 23. *Cross switch.*

Electromagnetic field processing

The collection of field processing modules is shown in Fig. 24. These modules are described in the Appendix and except for the source and bpm modules will not be described further here.

The source module is used to calculate the eigenmodes of ridge waveguide structures with the layer structure modeled by effective index. The source module takes the first slice (z = 0) of the geometry and material specification as input information. The panel for the source module is shown in Fig. 25. It is shown set to the numerical calculation mode. The initial power, calculation width, and mode selected for output are specified. The mode may also be read from a previously computed file by selecting file browser.

The calculation uses the finite difference method applied to the scalar wave equation to find eigenvalues of the lateral structure. Both guided modes and leaky modes are calculated. Leaky modes correspond to the modes of the calculation domain specified by the domain module. These are easily discernible from the bound modes, as shown in Fig. 26.

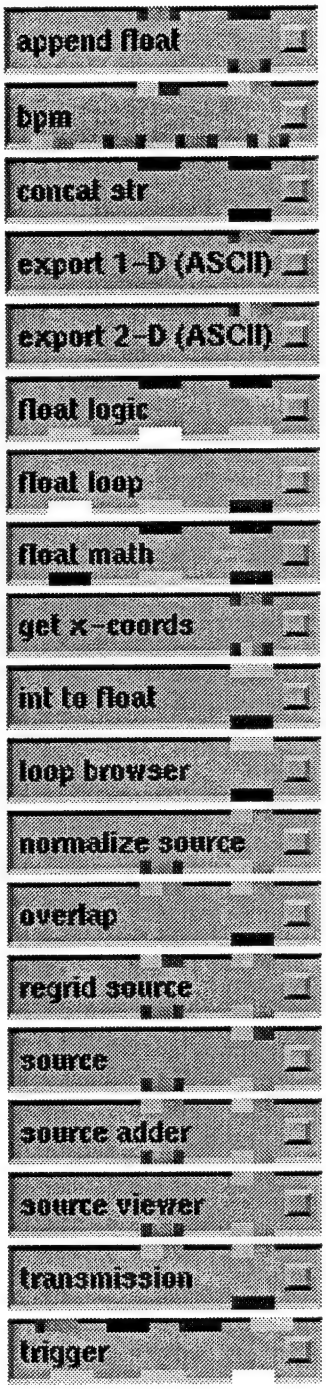


Figure 24. Field processing modules in the main menu.

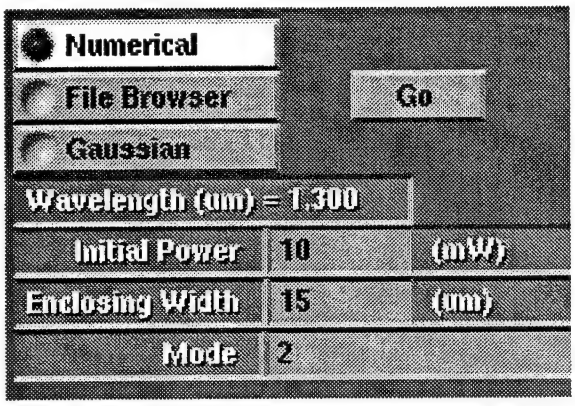


Figure 25. The source panel.

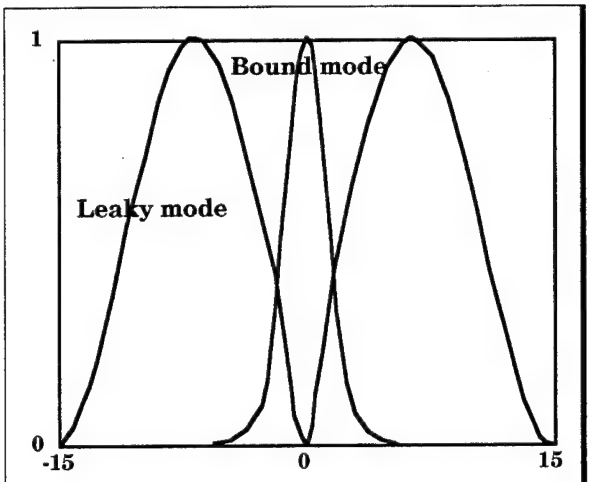


Figure 26. A comparison of a bound and leaky mode.

Computational Modules	
Module Name	**Description**
bpm	Solves the paraxial wave equation by the beam propagation method
Overlap	Calculates the overlap integral between two lateral field distributions
Regrid source	Places the source eigenmode onto a new numerical grid
Source	Calculates waveguide eigenmodes
Source adder	Combines one or more source modules into a multi-mode source
Transmission	A macro module that calculates the transmission coefficient

The bpm module is the center of the OEIC CAD system. We have endeavored to make this module general enough so that it can handle most problems that arise in this field. Although it is limited by the two-dimensional scalar wave equation it is capable of solving a wide variety of active and passive guided wave circuits.

The computational model is based upon the weak guide approximation in the lateral, x, direction and the effective index approximation in the transverse, y, direction; propagation is along the longitudinal, z, direction. The weak guide approximation requires variations in the dielectric to be relatively small, e.g., $\delta\left(n^2\right)/n_o^2 << 1$. The effective index approximation supports large index variations in the transverse direction, but also requires weak index variations in the lateral direction. Both of these approximations are well satisfied by the device structure under consideration. In addition we apply the transparent boundary condition at the lateral boundaries.

Under the weak guide approximation, if the electro-magnetic field does not change too rapidly in the direction of propagation then the Helmoltz equation reduces to the Fresnel equation, i.e.,

$$2jk_o n_o \frac{\partial \mathbf{E}}{\partial \mathbf{z}} = \left(\nabla_T^2 + k_o^2\left(n^2 - n_o^2\right)\right)\mathbf{E} .$$

This requirement puts significant restrictions on the applicability of the computational model. Specifically, it restricts applicability to systems that have no significant reflections in the direct path of propagation. Although methods exist for taking these reflections into account, they are not considered here.

In an active waveguide, where carriers are injected, the light and the carriers interact. This interaction is manifested as stimulated recombination and optical gain. Stimulated recombination detracts from the number of available carriers and its impact is modeled by a diffusion equation. Although diffusion is a 3-dimensional effect, we neglect diffusion effects in the transverse and longitudinal directions. It is justifiable to neglect the diffusion in the z direction since optical power and geometric structures change very slowly in that direction. Neglecting diffusion effects in the y direction is more a matter of convenience rather than easy justification. Note that the active region is limited to the intersection of the optical confinement region and the quantum well so that it is very small. We correct for ohmic spreading of the current in the cap layers between the contact and the active region so that $J(x)$ is specified at the active layer. Then, one-dimensional diffusion becomes a relatively accurate representation of the physics. Under these conditions, the diffusion equation is:

$$D\frac{d^2N}{dx^2} = -\frac{J(x)}{qd} + \frac{N}{\tau_{nr}} + BN^2 + CN^3 + \frac{g\Gamma|\mathbf{E}|^2}{\hbar\omega d},$$

where N is the density of free carriers, D is the carrier diffusion constant, d is the active layer thickness, $J(x)$ is the applied current density, τ_{nr} is the nonradiative recombination time, $B$is the spontaneous recombination coefficient, C is the Auger recombination coefficient, $g = aN\text{-}b$ is the optical power gain, a, is the gain coefficient, and b is the transparency loss, and Γ is the confinement factor.

The optical gain introduced by the carriers is modeled in the wave equation by introducing a complex effective index. The Fresnel equation in the presence of loss, α, is

$$2jk_o n_o \frac{\partial \mathbf{E}}{\partial \mathbf{z}} = \left(\nabla_T^2 + k_o^2\left(n^2 - n_o^2\right) - 2jk_o n_o \alpha\right)\mathbf{E}.$$

We note that in the absence of lateral and transverse effects that this equation reduces to $\partial\mathbf{E}/\partial\mathbf{z} = -\alpha\mathbf{E}$, indicating that for positive α, the field decays as it propagates in the z direction. We define the complex differential effective index

$$\Delta n^2 = n^2 - n_o^2 - 2j\frac{n_o}{k_o}\alpha + 2\Gamma\frac{n_o}{k_o}\left(aRN + j(aN - b)\right),$$

where R is the antiguiding parameter, which is introduced to model the depression in the real part of the effective index due to an increase in carriers in the confinement layer. The factor 2 is necessary to convert the effect of the power gain $aN\text{-}b$ into that of a voltage gain.

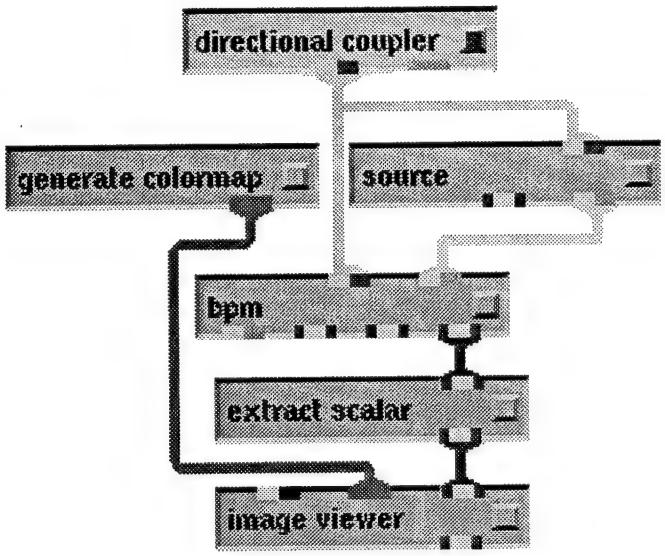


Figure 27. *A network for the simulation of a directional coupler.*

A typical network showing the use of the bpm module is shown in Fig. 27. The material and geometrical information have been encapsulated into the macro module labeled directional coupler. The source is used to provide the input eigenmode of the waveguide. The bpm panel is shown in Fig. 28. Note that it is a very simple panel since all the data has been accumulated by the preceding modules. All the user has to do is select the type of simulation. The linear bpm module is the fastest simulator since it ignores all the effects associated with pumping, antiguiding, diffusion, etc. The nonlinear mode incorporates all the physics previously discussed. The wide angle mode incorporates a higher order approximation to the scalar wave equation so that it can provide better solutions to problems that have significant off-axis activity. Finally, the file browser mode allows one to read in a previous simulation.

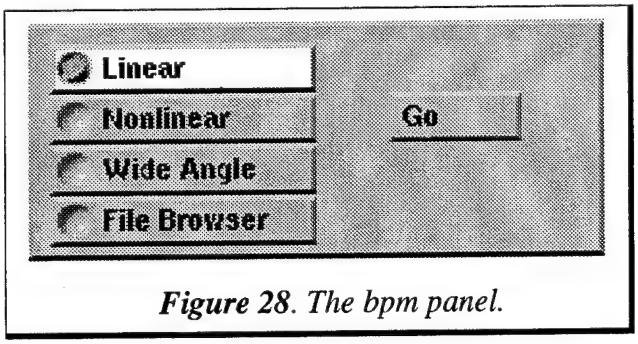


Figure 28. *The bpm panel.*

Visualization

AVS supplies a very large suite of modules to manipulate and visualize data. In this work we have concentrated on the conventional visualization techniques of images, surfaces, and line plots. The network in Fig. 27 shows a typical method for presenting computed data. The three modules, generate color map, extract scalar, and image viewer, create the image shown in Fig. 29. In this example two parallel waveguides are close enough to each other so that the field in one guide sufficiently overlaps the other guide

and power can be exchanged between guides. In this case, the guide length has been adjusted for about 3-dB coupling. Curved waveguide shifts are then used to separate the guides to stop the coupling process.

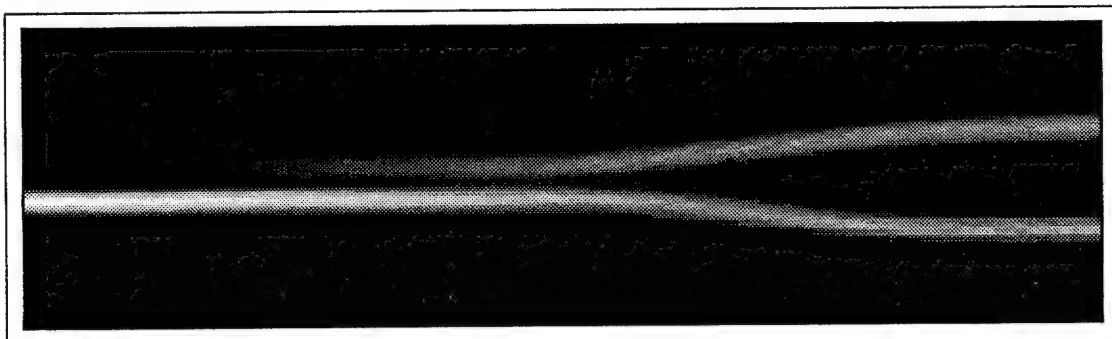


Figure 29. *An image of the optical field intensity.*

The generate color map module creates a data structure that is used to map field intensity to a gray level. The extract scalar module, whose panel is shown in Fig. 30, selects the type of data to be viewed (the bpm module generates several sets of data as shown in the panel). The image viewer renders the resulting image as shown in Figure 29. A surface plot of the effective index is shown in Fig. 31.

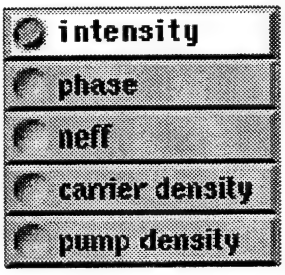


Figure 30. *The extract scalar panel.*

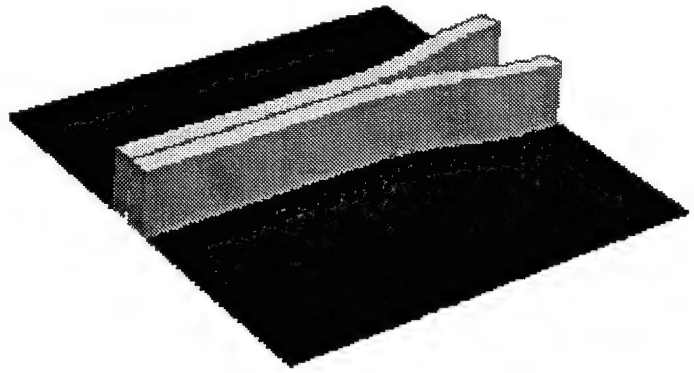


Figure 31. *3D plot of the effective index.*

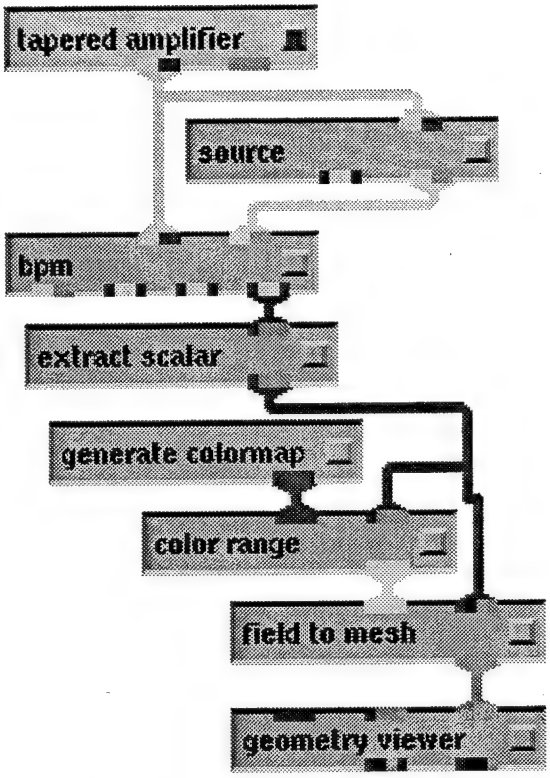


Figure 32. *Problem network for tapered laser amplifier.*

Figure 32 shows the problem network for simulating a linearly tapered laser amplifier. In this case the bpm module is set to nonlinear mode. The tapered amplifier macro module contains the material layer specifications and a waveguide geometry module. The waveguide module has the flare set to 1° and the current set to 10^{-4} A per square micron. The effective index is shown in Fig. 33, the light intensity is shown in Fig. 34, and the carrier density is shown in Fig. 35.

A waveguide mode corresponding to the mode of the initial width of the amplifier is injected at the left. The field interacts with the carriers and undergoes amplification and spreading as it progresses down the guide.

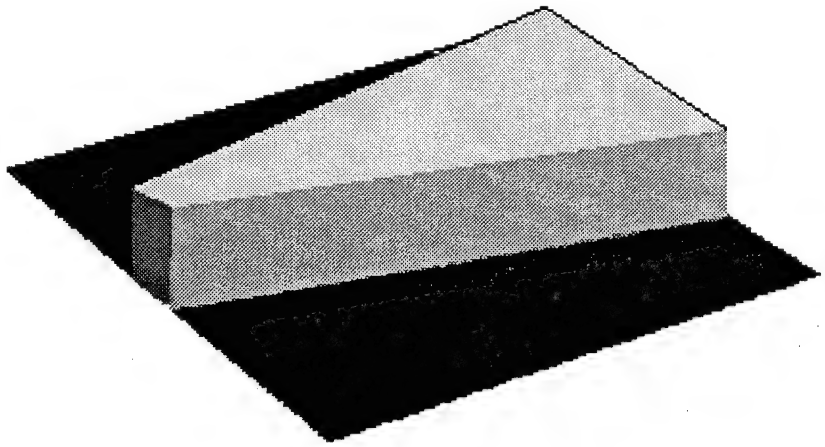


Figure 33. *Effective index in a tapered laser amplifier.*

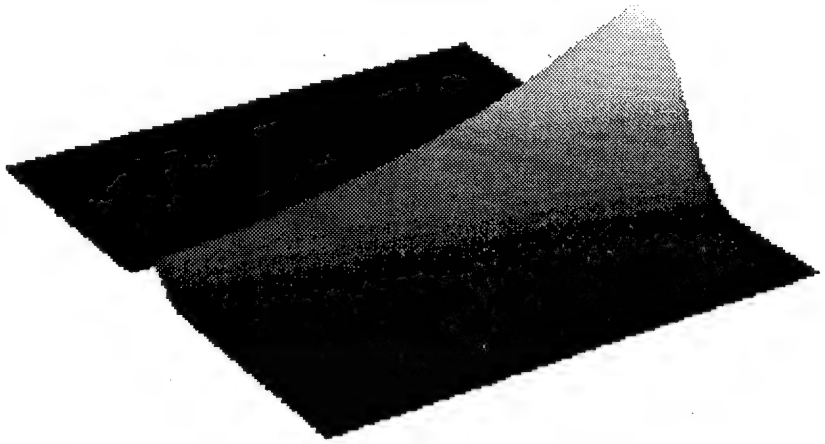


Figure 34. *Light intensity in the tapered laser amplifier.*

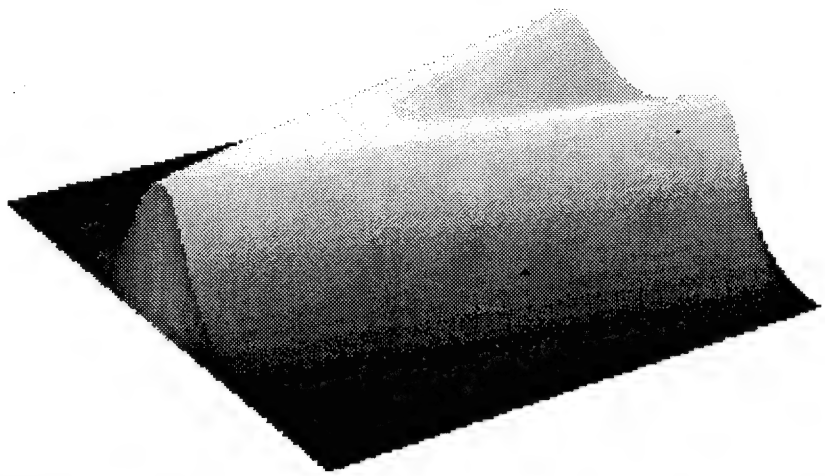


Figure 35. *Carrier density in the tapered laser amplifier.*

Since the light absorbs carriers during the amplification process, we see that the carriers are suppressed at the high light intensities. Notice also the peak carrier densities at the edges of the active region.

An important part of data visualization is the ability to vary parameters and observe the impact on the device. To this end we have added to the system a float loop module and other associated modules that can implement looping on any parameter in the network. AVS allows the user to create an input port for any module parameter. The float loop module is used to step through several values of the parameter. The float loop module also transmits a 'go' signal to the bpm module to start a new simulation. The

loop browser module automatically increments file names so that bpm results are saved for each variation. A typical network for such behavior is shown in Fig. 36. Here we have defined a beam splitter macro module composed of a curved y-split module followed by two straight waveguides, one connected to each branch of the splitter.

Parameters for this simulation are: simulation region = 20 μ by 1000 μ, splitter length = L, beam separation = s, ridge effective index = 3.45, wavelength = 0.9 μ, waveguide width = 3 μ, and background effective index = 3.4. A short beam splitter generates higher-order modes due to small radius of curvature. In severe cases (L = 250 μ and s = 12 μ), most of the beam departs the guiding structure and is not transferred to the straight section of waveguide. The results are shown in Fig. 37.

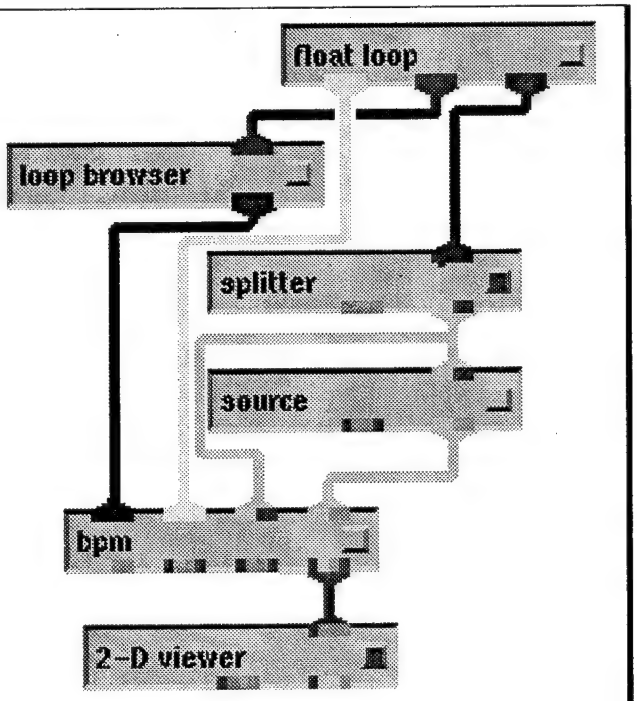


Figure 36. *A network incorporating looping.*

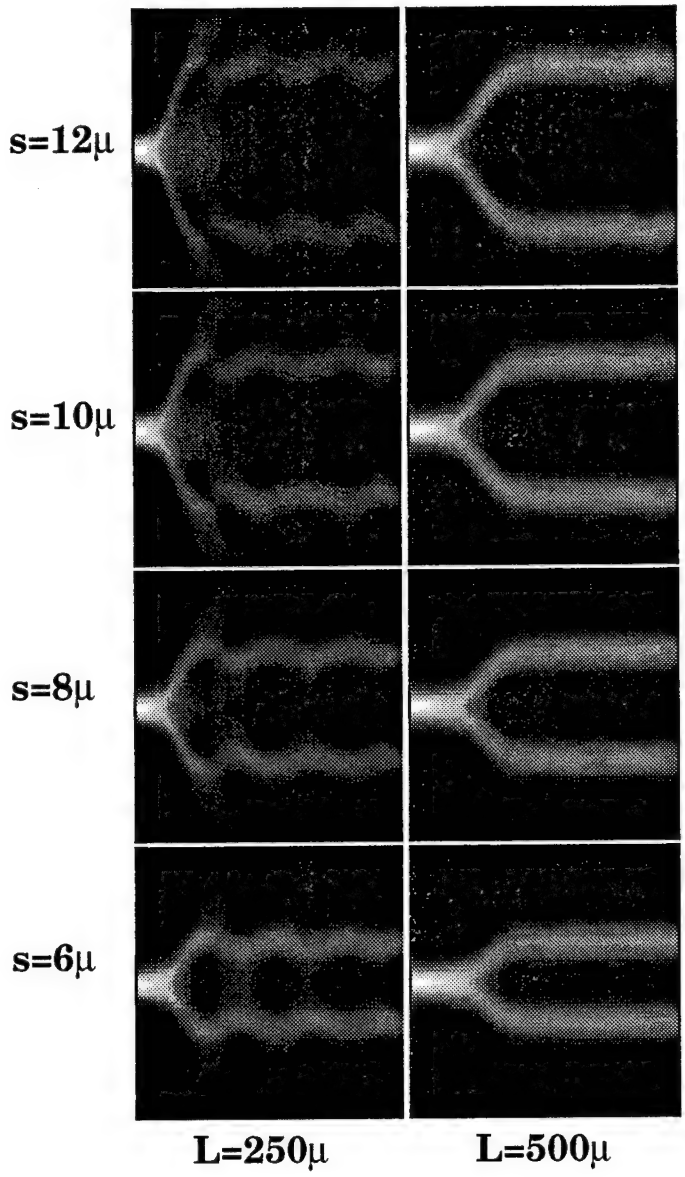


Figure 37. *Field intensity in a beam splitter as a function of splitter geometry.*

Task 3: Testing and Validation of the Optoelectronic CAD Tools

Objective

Sarnoff will test the simulators built in Task 2 and, as validation, compare the results to those obtained from existing stand-alone design tools. These simulators will be demonstrated to the government at Sarnoff. Simulated results will be compared to experimental results obtained at Sarnoff or other ULTRA contractors when possible.

Accomplishments

All results of bpm simulations were compared to our stand-alone simulator and we found no differences between the two versions. Comparisons between the bpm operating in different modes, e.g., linear, nonlinear, and wide angle showed no significant difference in cases where no difference should be present, e.g., unpumped systems with fields staying within 5° of the longitudinal axis.

We are working with JPL on a WDM circuit. Simulations have been performed and the results sent to JPL. JPL has fabricated masks based on these simulations. The devices are now being fabricated.

Task 4: Program Management

Objective

Sarnoff will designate a Program Manager (PM) to manage and control the contractor efforts for all tasks under this contract. The PM shall be the single point of contact for all managerial matters and shall be responsible for all efforts necessary for the efficient and effective performance of this contract.

Accomplishments

The program was effectively managed and all objectives delimited above in italics were accomplished.

Deliverables

The following items will be delivered to ARPA during this program.

- The simulation modules developed under Task 2 that can be run under the selected COTS SVE software package
- Final report describing the design and operation of the optoelectronic simulators

4. Results and Conclusions

Sarnoff has created a flexible, comprehensive system for OEIC CAD. We have demonstrated that stand-alone simulators such as our bpm and eigenmode solver can be modified and incorporated into a COTS visualization system without loss of efficiency and correctness. With the careful design of data structures and modules we were able to synthesize a system that can be rapidly deployed to solve very complicated problems with a minimum of data input and user interaction.

The key factor in developing a system that exhibits data inheritance, data sharing, flexibility, and extensibility is to carefully design the data structures that flow through a problem network. We have found the best data structure design is arrived at by emulating the real-world fabrication and testing process, e.g., material modules emulate growth, geometry modules emulate layout, and processing modules emulate testing.

Another important aspect of this work is the recognition of the need to know what physical properties are necessary to perform successful simulations along with their interrelationships. This information creates a map from which the data structures can be defined.

This OEIC CAD system should open the door to greater opportunity for researchers to share their theoretical developments, check the accuracy of their simulation code against others, and collaborate on the design of OEIC.

Appendix

Module Documentation

Introduction

This Appendix contains a listing of the on-line module help files. These help files are accessible by clicking in an AVS panel. They contain parameter definitions, input and output definitions, and example problem networks.

NAME

append float - append new data points to an existing field or create a new field if the input field port is not connected.

SUMMARY

Name	append float	
Availability	OEIC module library	
Type	processing	
Inputs	field 1-D uniform float (OPTIONAL)	
	float (1-8 inputs depending on # of Ports parameter)	
Outputs	field 1-D uniform float	
Parameters	Name	Type
	Reset	oneshot
	Validate	boolean
	Ports	typein_integer
	Label	string (1-8 depending on # of Ports parameter)

DESCRIPTION

The append float module allows the user to append new data to the end of an existing field or to create a new field of data if the field input port is not connected. The number of vectors (ports) is entered as a parameter or read from the existing field. Labels for each vector can be typed into the appropriate string parameter. A new data set will be appended when any of the input ports data has changed, unless the 'validate inputs' option is selected, then data will only be appended when all of the input ports have changed (when using the 'validate inputs' option it is an error for an input to change twice before all other inputs have changed).

INPUTS

field 1-D uniform float (OPTIONAL)

If this port is connected the new data sets will be appended onto the end of this field. The number of input ports is the number of vectors in the field, and the vector labels will be those saved in the field.

float (1-8 inputs depending on # of Ports parameter)

These ports are used for the new data, the number of ports is determined from the field input or typed in the '# of ports' parameter if the field input is not connected.

PARAMETERS

Reset

Resets the data count of the output field, if the field input port is not connected this value will be reset to zero, if the field input port is connected this value will be reset to the number of data points in the input field.

Validate

This option toggles the 'validate inputs' mode. When this mode is disabled a new data set will be appended when any of the input ports changes value, when enabled data will only be appended when all input ports have changed value (when enabled it is an error for an input to change twice before all other inputs have changed).

Ports

This is the number of input ports made available for appending new data. If the field input port is connected the number of ports is set to the number of vectors in the input field, otherwise it is a user entered parameter.

Label [0-7]

Labels for each vector in the output field. If the field input port is connected the labels are defaulted to the labels used in the input field, otherwise they are defaulted to 'channel [0-7]'.

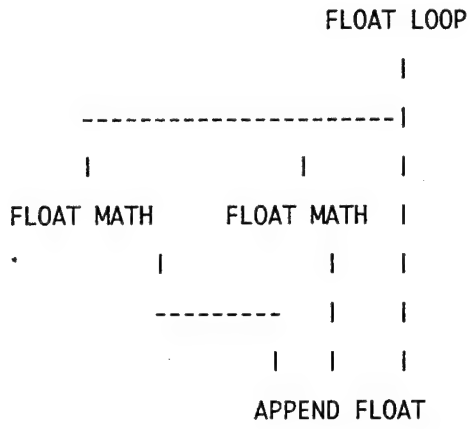
OUTPUTS

field 1-D uniform float

The field output contains the old field data (if connected) and the new data appended to the end.

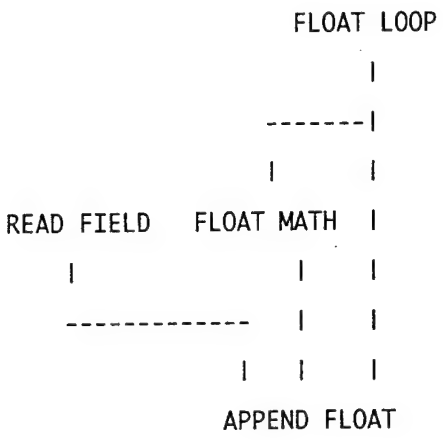
EXAMPLE 1

This example displays an append float module being used to create a new field of data sets:



EXAMPLE 2

This example displays an append float module being used to append new data sets to an existing field:



RELATED MODULES

float loop
read field

Release 1 OEIC append float()

NAME

axis viewer - view the geometry axis data port parameters

SUMMARY

Name axis viewer

Availability OEIC module library

Type geometry

Inputs axis data (user data type)

Outputs axis data (user data type)

Parameters

Name	Type
Id	string
Width	string
Length	string
Nx	string
Nz	string
Zstep	string
Alpha	string
Wavelength	string
Neff	string
Loss	string
Corr Length	string
Fluctuation	string
dn/dx	string
dn/dz	string
dloss/dx	string
dloss/dz	string
width	string
x	string
z	string

DESCRIPTION

The axis viewer module allows the user to view the geometry parameters and axis data coming from an axis data (user data type) output port.

INPUTS

Axis data (axis data user data type)

The axis data input usually comes from a geometry module (domain, waveguide, ...) and contains the geometry domain parameters and the axis specification for the next geometry module.

PARAMETERS

Id

Displays the module id of the geometry.

Width

Displays the domain width of the geometry.

Length

Displays the domain length of the geometry.

Nx

Number of grid points in the transverse dimension.

Nz

Number of grid points in the longitudinal dimension.

Zstep

Length of propagation step for the numerical solution.

Alpha

Sets a nonuniform x grid. If Alpha < 1, the x grid is denser at the center; if Alpha > 1, the x grid is denser at the edges.

Wavelength

Wavelength of the propagating field.

Neff

Index of refraction.

Loss

Loss.

Corr Length

Physical extent of nonuniformity correlation.

Fluctuation

Magnitude of nonuniformity fluctuation.

dn/dx

Index gradient in x direction.

dn/dz

Index gradient in z direction.

dloss/dx

Grating out coupling coefficient.

dloss/dz

Grating wavenumber.

width

Starting width of next geometry.

x

X grid starting position for next geometry.

z

Z grid starting position for next geometry.

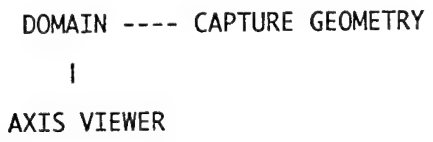
OUTPUTS

Axis data (axis data user data type)

The axis data output contains the geometry domain parameters and the axis specification for the next geometry module, identical to the input axis data.

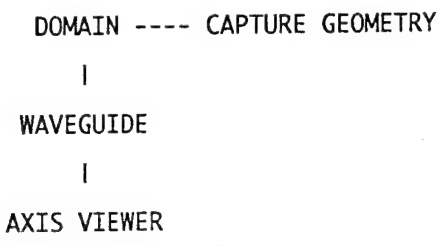
EXAMPLE 1

This example displays an axis viewer module being used to view the geometry domain parameters and the starting axis information of the geometry:



EXAMPLE 2

This example displays an axis viewer module being used to view the geometry domain parameters and the axis information for the next geometry:



RELATED MODULES

domain

capture geometry

Release 1 OEIC axis viewer()

NAME

bpm - calculate beam propagation

SUMMARY

Name bpm

Availability OEIC module library

Type processing

Inputs geomdata (geometry user data type)
srcdata (source user data type)

Outputs field 2-D 2-space float
field 2-D 2-space float
field 2-D 2-space float
srcdata (source user data type)

Parameters

Name	Type
Go	oneshot
Go Linear	oneshot
Filename	file_browser

DESCRIPTION

The bpm module calculates the beam propagation.

INPUTS

Geometry data (geomdata user data type)

The geometry input usually comes from the capture geometry module and is a geometry specification using the geomdata user data type.

Source data (srcdata user data type)

The srcdata input contains the source information (wavelength, power, mode shift, filename) and the normalized solution mode data (normalized to peak value of 1). This data usually comes from the source module.

PARAMETERS

Go

Starts the non-linear beam propagation calculations, must be used if there is pumping in any of the geometry pieces.

Go Linear

Starts the linear beam propagation calculations, this method is much faster than the non-linear case but does not take into account any pumping of the geometry.

Filename

Allows the user to choose a directory and base filename for the bpm output. The following files will be created:

```
filename.out     ! output file for runtime messages
filename.xz.fld  ! field file containing the 2-D data
filename.z.fld   ! field file containing the 1-D z-data
filename.x.fld   ! field file containing the 1-D x-data
```

OUTPUTS

field 2-D 2-space float

This output port contains the 2-D data (intensity, phase, neff, carrier density, pump density).

field 2-D 2-space float

This output port contains the 1-D z-data (power, asymmetry, beam width, real E @ z=0, imag E @ z=0).

field 2-D 2-space float

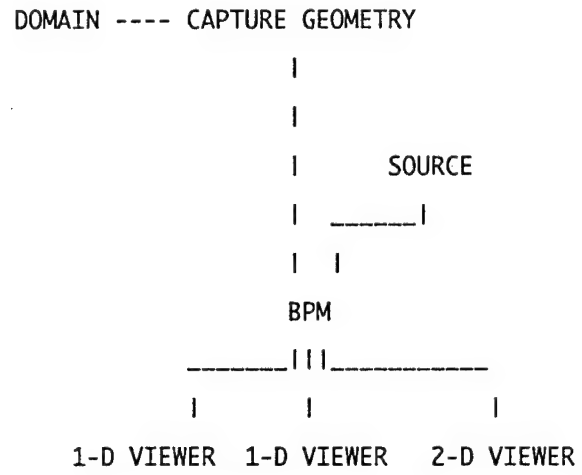
This output port contains the 1-D x-data (real E, imag E, intensity, phase).

Source data (srcdata user data type)

The srcdata output contains the source information (wavelength, power, mode shift, filename) and the normalized solution mode data (normalized to peak value of 1). This data can be passed to another bpm module to continue the propagation.

EXAMPLE

This example displays a typical use of the bpm module:



RELATED MODULES

source

capture geometry

2-D viewer

1-D viewer

NAME

loop browser - filename browser, appends integer extension '.XXX' if integer input port connected

SUMMARY

Name	loop browser
Availability	OEIC module library
Type	processing
Inputs	integer
Outputs	string
Parameters	Filename file_browser

DESCRIPTION

The loop browser module allows the user to select a directory and filename to be output to a downstream module. If the integer input port is connected a '.XXX' file extension is appended. This module is handy for reading or writing a sequence of files with the '.XXX' extension.

INPUTS

integer

The integer value to be appended to the filename.

PARAMETERS

Filename

The file browser used to select the directory and base filename.

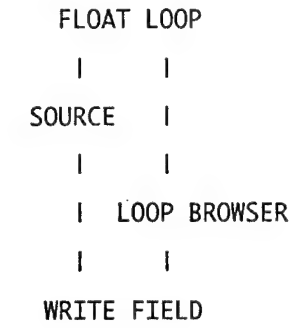
OUTPUTS

string

The absolute path name of the selected file with extension if needed.

EXAMPLE

This example displays a loop browser module being used to write a sequence of output files with the same base filename and a '.XXX' file extension:



RELATED MODULES

animated float

float loop

NAME

capture geometry - geometry wrapper for collecting geometry

SUMMARY

Name	capture geometry	
Availability	OEIC module library	
Type	geometry	
Inputs	objectData	(geometry object user data type) [INVISIBLE]
	axisData	(geometry axis user data type)
Outputs	geomdata	(geometry user data type)
Parameters	NONE	

DESCRIPTION

The capture geometry module collects all geometry pieces and combines and passes them to the geometry viewer.

INPUTS

objectData (geometry object user data type) [INVISIBLE]

The geometry object information passed directly from the geometry module through an invisible connection.

axisData (geometry axis user data type)

The geometry axis information which passes the geometry axis data.

PARAMETERS

NONE

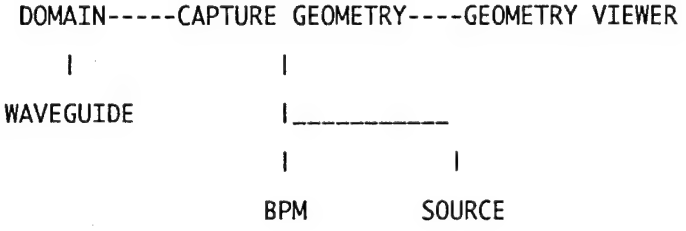
OUTPUTS

geomdata (geometry user data type)

The geometry object and axis information along with the material substrate parameters. This data is usually passed to the bpm and source modules .

EXAMPLE

This example displays a capture geometry module being used in a geometry:



RELATED MODULES

domain
geometry viewer
bpm
source
axis viewer
object viewer
region viewer

Release 1 OEIC capture geometry()

NAME

composite material - emulate a composite material

SUMMARY

Name composite material

Availability OEIC module library

Type material

Inputs matdata (material user data type) [OPTIONAL]

Outputs compdata (composite material user data type)

Parameters

Name	Type	Default	Units
Wavelength	float	0.9	um
Neff	float	0.0	
Loss	float	0.0	1/um
Corr Length	float	0.0	um
Fluctuation	float	0.0	
dn/dx	float	0.0	1/um
dn/dz	float	0.0	1/um
dloss/dx	float	0.0	1/um2
dloss/dz	float	0.0	1/um2
Random	boolean	false	
Linear	boolean	false	
GA	float	1E-8	um2
GB	float	0.0	um2
AFC	float	10.0	
AGD	float	2.0	
GMA	float	2.29E-2	
TCAP	float	1.5	um
ABS	float	1E-2	1/um
LABS	float	6.0	um

DIFF	float	1E9	um2/sec
TNR	float	5E-9	sec
BSR	float	26.52	um3/sec
AUGER	float	0.0	um6/sec
ALT	float	0.008	um

DESCRIPTION

The composite material module emulates a material composite. Instead of building a material structure and calculating the material parameters using the modeig module, the user can enter all relevant data using the composite material module. By connecting the material data input port to the modeig module output port this module can be used to view the material parameters that the modeig module calculated.

INPUTS

material data (optional; matdata user data type)

The material data input usually comes from the modeig modules material output port. When this port is connected all the composite material parameter values are defaulted to the values calculated by the modeig module.

PARAMETERS

Wavelength

Wavelength of the propagating field (microns).

Neff

Index of refraction.

Loss

Loss (inverse microns).

Corr Length

Physical extent of nonuniformity correlation (microns).

Fluctuation

Magnitude of nonuniformity fluctuation.

dn/dx

Index gradient in x direction (inverse microns).

dn/dz

Index gradient in z direction (inverse microns).

dloss/dx

Grating out coupling coefficient (inverse square microns).

dloss/dz

Grating wavenumber (inverse square microns).

Random

Enables use of the random nonuniformity parameters.

Linear

Enables use of the linear nonuniformity parameters.

GA

Gain parameter (g = GA * N - GB, where N = carrier density)
units = square microns.

GB

Gain parameter (g = GA * N - GB, where N = carrier density).
units = square microns.

AFC

Unpumped loss saturation factor.

AGD

Antiguiding factor.

GMA

Confinement factor.

TCAP

Thickness between the active layer and the surface (microns).

ABS

Boundary absorption parameter (inverse microns).

LABS

Boundary absorption length (microns).

DIFF

Carrier diffusion coefficient (square microns per second).

TNR

Nonradiative recombination time.

BSR

Spontaneous recombination coefficient (cubed microns per second).

AUGER

Auger recombination coefficient (um6 per second).

ALT

Active layer thickness (microns).

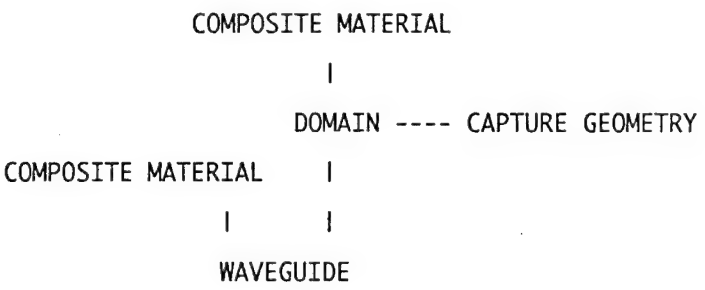
OUTPUTS

Composite Material data (compdata user data type)

The compdata output contains the composite material information and is used as input for a geometry.

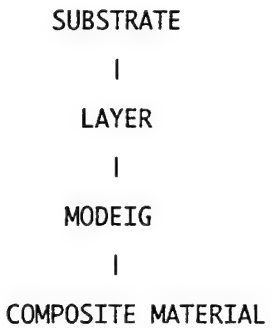
EXAMPLE 1

This example displays a composite material module being used to input the material parameters to the geometry elements:



EXAMPLE 2

This example displays a composite material module being used to capture and view the material parameters being passed from the modeig module:



RELATED MODULES

modeig
substrate
layer

SEE ALSO

The demos FOAL, COUPLER, and WAVE, as well as others demonstrate the composite material module.

Release 1 OEIC composite material()

NAME

curved waveguide - create a geometry that curves the waveguide by the arc length parameter

SUMMARY

Name curved waveguide

Availability OEIC module library

Type geometry

Inputs

axisData	(geometry axis user data type)
compdata	(composite material user data type)

Outputs

objectData	(geometry object user data type)	[INVISIBLE]
axisData	(geometry axis user data type)	

Parameters

Name	Type	Default	Units
Title	string_block	---	---
Change Width	float	0.0	um
Length	float	100.0	um
Arc Length	float	0.0	um
Offset X-axis	float	0.0	um
Offset Z-axis	float	0.0	um
Curr. Density	float	0.0	A/um2
Flip Horizontal	boolean	false	---
Flip Vertical	boolean	false	---

DESCRIPTION

The curved waveguide module creates a geometry that curves the waveguide by the arc length parameter. The waveguide can then be flipped in the horizontal and vertical directions.

Note: The capture geometry module must be present when this module is instantiated so the objectData invisible connection can be made.

INPUTS

axisData (geometry axis user data type)

The geometry axis data, usually comes from a capture geometry module.

compdata (composite material user data type)

The composite material input, usually comes from a composite material module.

PARAMETERS

Title

Displays pertinent information about the waveguide shift geometry and material parameters.

Change Width

Change in the width of the waveguide from the preceding waveguide (microns).

Length

Length of the waveguide (microns).

Arc Length

Arc length of the curved waveguide (microns).

Offset X-axis

X-directional offset of the waveguide from the preceding waveguide (microns).

Offset Z-axis

Z-directional offset of the waveguide from the preceding waveguide (microns).

Curr. Density

Current density (pumping) of the waveguide (A/um2).

Flip Horizontal

Toggles the horizontal flip of the curved waveguide.

Flip Vertical

Toggles the vertical flip of the curved waveguide.

OUTPUTS

objectData (geometry object user data type) [INVISIBLE]

The geometry object information passed directly to the capture geometry module through an invisible connection.

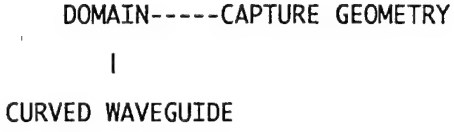
Note: The capture geometry module must be present when this module is instantiated so this invisible connection can be made.

axisData (geometry axis user data type)

The geometry axis information which passes the current axis data to the succeeding geometry module.

EXAMPLE

This example displays a curved waveguide module being used in a geometry:



RELATED MODULES

domain
capture geometry
waveguide
waveguide shift
axis viewer
object viewer

NAME

domain - starting point for building a new geometry

SUMMARY

Name domain

Availability OEIC module library

Type geometry

Inputs compdata (composite material user data type)

Outputs axisData (geometry axis user data type)

Parameters

Name	Type	Default	Units
Title	string_block	---	---
Domain Width	float	60.0	um
Domain Length	float	500.0	um
Nx	integer	181	---
Nz	integer	101	---
Min Z-step	float	0.0	um
Alpha	float	0.0	---
refresh	oneshot	---	---

DESCRIPTION

The domain module creates the foundation for a new geometry, it automatically instantiates a capture geometry and a geometry viewer module.

INPUTS

compdata (composite material user data type)

The composite material input, usually comes from a composite material module.

PARAMETERS

Title

Displays pertinent information about the waveguide shift geometry and material parameters.

Domain Width

The domain width of the geometry (microns).

Domain Length

The domain length of the geometry (microns).

Nx

Number of grid points in the transverse dimension.

Nz

Number of grid points in the longitudinal dimension.

Min Z-step

Length of propagation step for the numerical solution (microns).

Alpha

Sets a nonuniform x grid. If Alpha < 1, the x grid is denser at the center; if Alpha > 1, the x grid is denser at the edges.

Refresh

Refreshes the geometry viewer display, used when internal geometry pieces are removed from the whole geometry.

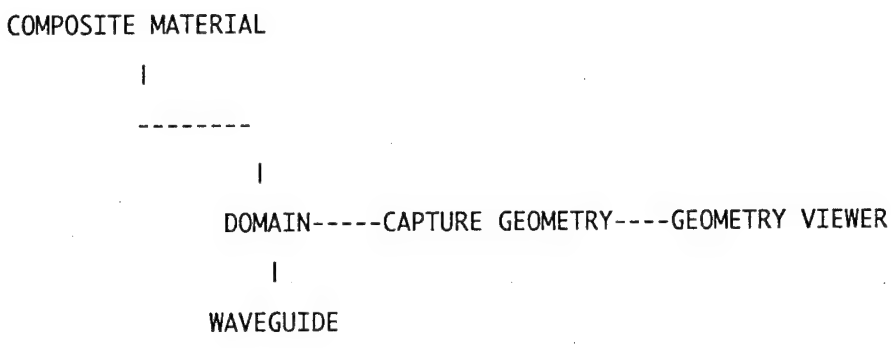
OUTPUTS

axisData (geometry axis user data type)

The geometry axis information which passes the initial axis data to the succeeding geometry module.

EXAMPLE

This example displays a domain module being used in a geometry:



RELATED MODULES

capture geometry
waveguide
composite material
geometry viewer
axis viewer

NAME

export 2-D - takes a 2-D scalar field input and creates an ASCII text file

SUMMARY

Name export 2-D

Availability OEIC module library

Type processing

Inputs field 2-D scalar

Outputs text file (.txt)

Parameters

Name	Type
Special	boolean
Filename	file_browser

DESCRIPTION

The export 2-D module takes a 2-D scalar field input and creates an ASCII text file useful for importing into another data visualization program. The ASCII SPECIAL option allows for direct import into Spyglass Transform. The output file will have the extension '.txt'.

INPUTS

field 2-D scalar

The data to be written to the ascii file.

PARAMETERS

Special

Toggles the ASCII SPECIAL file format option.

Filename

Allows the user to select a directory and filename for the output file.

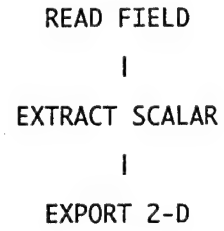
OUTPUTS

text file (.txt)

The output file will be written to the directory selected using the file browser.

EXAMPLE 1

This example displays an export 2-D module being used write a set of data to disk:



RELATED MODULES

extract scalar
read field

Release 1 OEIC export 2-D()

NAME

export 1-D - takes a 1-D field input and creates an ASCII tab-delimited columnar text file, with column labels optional.

SUMMARY

Name	export 1-D	
Availability	OEIC module library	
Type	processing	
Inputs	field 1-D	
Outputs	text file (.txt)	
Parameters	Name	Type
	Titles	boolean
	Filename	file_browser

DESCRIPTION

The export 1-D module takes a 1-D field input and creates an ASCII tab-delimited columnar text file useful for importing into other plotting packages. The 'Titles' option will output the labels for each column. The output file will have the extension '.txt'.

INPUTS

field 1-D

The data to be written to the ascii file.

PARAMETERS

Titles

Toggles the column labels output option.

Filename

Allows the user to select a directory and filename for the output file.

OUTPUTS

text file (.txt)

The output file will be written to the directory selected using the file browser.

EXAMPLE 1

This example displays an export 1-D module being used write a set of data to disk:

READ FIELD

|

EXPORT 1-D

RELATED MODULES

export 2-D
read field
get x-coords
1-D viewer
graph viewer

NAME

float loop - identical to the animated float module except for the additional integer and oneshot output ports.

SUMMARY

Name	float loop
Availability	OEIC module library
Type	processing
Inputs	NONE
Outputs	float integer oneshot
Parameters	See the animated float module documentation

DESCRIPTION

The float loop module is identical to the animated float module except for the additional integer and oneshot output ports. The integer output port represents the step number and the oneshot port can be used to trigger downstream modules.

INPUTS

NONE

PARAMETERS

See the animated float module documentation

OUTPUTS

float

The floating point output value.

integer

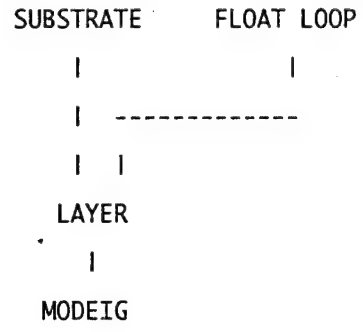
The step number.

oneshot

Trigger output.

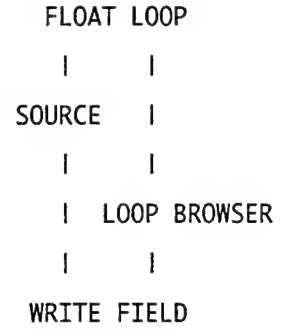
EXAMPLE 1

This example displays a float loop module being used to loop a material parameter:



EXAMPLE 2

This example displays a float loop module being used to loop a parameter and write the sequence of output files with the same base filename and a '.XXX' file extension:



RELATED MODULES

animated float

float

loop browser

Release	1	OEIC	float loop()

NAME

float logic - performs logical operations on two floating point inputs

SUMMARY

Name float logic

Availability OEIC module library

Type processing

Inputs float
float [OPTIONAL]

Outputs integer
oneshot
boolean

Parameters

Name	Type
Display	string_block
Function	choice { =, !=, x, y, >, >=, <, <= }
Invert	boolean

DESCRIPTION

The float logic module performs the selected logical operation on the two floating point inputs (the second float input is optional for some operations). The inputs, operation, and result is displayed in a string on the control panel. The logical result can be inverted (0->1 or 1->0) if the 'invert' option is selected. The result is output as an integer, oneshot, and a boolean for the users convenience.

Note: The 'invert' option will only invert the output data, not the result shown in the 'Display' string.

INPUTS

float

The floating point input that corresponds to the x operand.

float [OPTIONAL]

The floating point input that corresponds to the y operand.

PARAMETERS

Display

A string_block that displays the inputs, operation, and result. Note: The 'invert' option has no effect on the result displayed in this string.

Function

The logical function to perform on the inputs.

Invert

Invert the logical result (zero will be inverted to one and one will be inverted to zero). Note: Enabling this option will only invert the output data, not the result shown in the 'Display' string.

OUTPUTS

integer

The result of the logical operation (inverted if desired).

oneshot

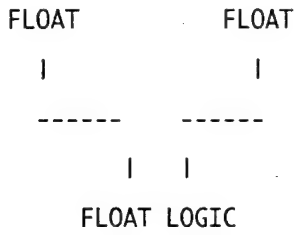
The result of the logical operation (inverted if desired).

boolean

The result of the logical operation (inverted if desired).

EXAMPLE

This example displays a float logic module being used to perform a logical operation on two float inputs:



RELATED MODULES

float
float math

Release	1	OEIC	float logic()

NAME

float math - performs arithmetic operations on two floating point inputs

SUMMARY

Name: float math

Availability: OEIC module library

Type: processing

Inputs: float
float [OPTIONAL]

Outputs: float
integer
string

Parameters:

Name	Type
Display	string_block
Function	choice { +, -, *, /, +/-, abs, sqr, sqrt, y^x, y^1/x, 1/x, cbrt, sin, cos, tan, x, asin, acos, atan, y, log, ln, e^x, 10^x, min, max }
DRG	boolean

DESCRIPTION

The float math module performs the selected arithmetic operation on the two floating point inputs (the second float input is optional for some operations). The inputs, operation, and result is displayed in a string on the control panel. The units of the trigonometric functions can be toggled between degrees and radians using the 'DRG' button. The result is output as a float, integer (result truncated), and a string (float) for the users convenience.

INPUTS

float

The floating point input that corresponds to the x operand.

float [OPTIONAL]

The floating point input that corresponds to the y operand.

PARAMETERS

Display

A string_block that displays the inputs, operation, and result.

Function

The arithmetic function to perform on the inputs.

DRG

Toggle the units for the trigonometric functions between degrees and radians.

OUTPUTS

float

The floating point result of the arithmetic operation.

integer

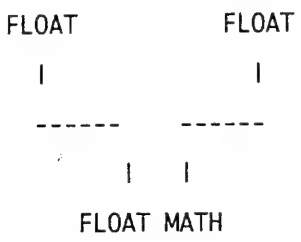
The truncated result of the arithmetic operation.

string

The floating point result of the arithmetic operation as a string.

EXAMPLE

This example displays a float math module being used to perform an arithmetic operation on two float inputs:



RELATED MODULES

float
float logic

NAME

graded layer - generate the material graded layer parameters

SUMMARY

Name graded layer

Availability OEIC module library

Type material

Inputs laydata (layer user data type)

Outputs laydata (layer user data type)

Parameters

Name	Type	Default	Units
Layer Name	string	NULL	
Material	choice	Substrate	
Steps	integer	10	
First Alum	float	0.0	%
Last Alum	float	0.0	%
First Q	float	0.0	
Last Q	float	0.0	
First Index	float	0.0	
Last Index	float	0.0	
Loss	float	0.0	1/um
Thickness	float	0.0	um
Doping	float	0.0	1/um3

DESCRIPTION

The graded layer module allows the user to enter the number of steps and the first and last parameters used to calculate the index of refraction for each layer. The thickness of each layer is the total thickness divided by the number of steps.

PARAMETERS

Layer Name

Identifying name of the graded layer, each slice will have an integer identifier appended to the end corresponding to the slice number in the grade.

Material

Material composition of the layer. Upon connection to the preceding module this value will be defaulted to the same material as the substrate {GaAlAs, InGaAsP, Other}.

First Aluminum

Percentage of aluminum in GaAlAs material of the first slice in the graded layer, it is used to calculate the index of refraction. All intermediate slice values will be incremented by the aluminum percentage range divided by the number of steps. This parameter will only be visible when GaAlAs is selected as the layer material. (Range = 0-100%)

Last Aluminum

Percentage of aluminum in GaAlAs material of the last slice in the graded layer, it is used to calculate the index of refraction. All intermediate slice values will be incremented by the aluminum percentage range divided by the number of steps. This parameter will only be visible when GaAlAs is selected as the layer material. (Range = 0-100%)

First Q

Q value of InGaAsP material of the first slice in the graded layer, it is used to calculate the index of refraction. All intermediate slice values will be incremented by the Q value range divided by the number of steps. This parameter will only be visible when InGaAsP is selected as the layer material. (Range = 0-1.55)

Last Q

Q value of InGaAsP material of the last slice in the graded layer, it is used to calculate the index of refraction. All intermediate slice values will be incremented by the Q value range divided by the number of steps. This parameter will only be visible when InGaAsP is selected as the layer material. (Range = 0-1.55)

First Index

Index of refraction of the first slice in the graded layer. All intermediate slice values will be incremented by the index range divided by the number of steps. This parameter will only be visible when 'Other' is selected as the layer material.

Last Index

Index of refraction of the last slice in the graded layer. All intermediate slice values will be incremented by the index range divided by the number of steps. This parameter will only be visible when 'Other' is selected as the layer material.

Loss

Loss of each slice in the graded layer (inverse microns).

Thickness

Total thickness of graded layer, the thickness of each slice is the total thickness divided by the number of steps (microns).

Doping

Doping of each slice in the graded layer (inverse cubed microns).

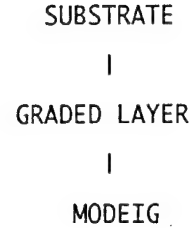
OUTPUTS

Layer data (laydata user data type)

The layer data output contains the substrate parameters and the parameters for each layer in the structure.

EXAMPLE 1

This example displays the graded layer module being used in a composite material:



RELATED MODULES

layer
layer viewer
substrate
modeig
composite material

Release 1 OEIC graded layer()

NAME

int to float - converts an integer input to a floating point output

SUMMARY

Name	int to float
Availability	OEIC module library
Type	processing
Inputs	integer
Outputs	float
Parameters	None

DESCRIPTION

The int to float module converts an integer input to a floating point output.

INPUTS

integer

The integer input.

PARAMETERS

NONE

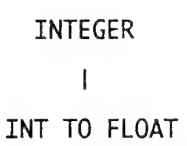
OUTPUTS

float

The floating point output.

EXAMPLE

This example displays an int to float module being used to convert an integer value to a floating point value:



RELATED MODULES

integer
float

NAME

layer - generate the material layer parameters

SUMMARY

Name layer

Availability OEIC module library

Type material

Inputs laydata (layer user data type)

Outputs laydata (layer user data type)

Parameters

Name	Type	Default	Units
Layer Name	string	NULL	
Material	choice	Substrate	
Aluminum	float	0.0	%
Q	float	0.0	
Index	float	0.0	
Loss	float	0.0	1/um
Thickness	float	0.0	um
Doping	float	0.0	1/um3

DESCRIPTION

The layer module allows the user to enter the material layer parameters.

PARAMETERS

Layer Name

Identifying name of the layer.

Material

Material composition of the layer. Upon connection to the substrate module this value will be defaulted to the same material as the substrate {GaAlAs, InGaAsP, Other}.

Aluminum

Percentage of aluminum in GaAlAs material used to calculate the index of refraction. This parameter will only be visible when GaAlAs is selected as the layer material. (Range = 0-100%)

Q

Q value of InGaAsP material used to calculate the index of refraction. This parameter will only be visible when InGaAsP is selected as the layer material. (Range = 0-1.55)

Index

Index of refraction. This parameter will only be visible when Other is selected as the layer material.

Loss

Loss (inverse microns).

Thickness

Thickness of layer (microns).

Doping

Doping of layer (inverse cubed microns).

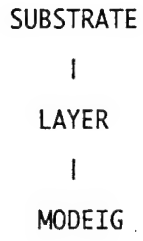
OUTPUTS

Layer data (laydata user data type)

The layer data output contains the substrate parameters and the parameters for each layer in the structure.

EXAMPLE 1

This example displays the layer module being used in a composite material:



RELATED MODULES

layer viewer
substrate
modeig
composite material

NAME

layer viewer - view the laydata port parameters

SUMMARY

Name layer viewer

Availability OEIC module library

Type material

Inputs laydata (user data type)

Outputs laydata (user data type)

Parameters

Name	Type
Layer Count	string
Substrate	choice
Wavelength	string
Corr Length	string
Fluctuation	string
dn/dx	string
dn/dz	string
dloss/dx	string
dloss/dz	string
Layer#	typein integer
Layer Name	string
Index	string
Loss	string
Thickness	string
Doping	string

DESCRIPTION

The layer viewer module allows the user to view the substrate parameters and the individual layer data coming from a laydata (user data type) output port. By typing in the desired layer number the corresponding layer information will be displayed.

INPUTS

Layer data (laydata user data type)

The layer data input usually comes from a layer or substrate module and contains the substrate parameters and the parameters for each layer in the structure.

PARAMETERS

Layer Count

Displays the number of layers.

Substrate

The substrate material type.

Wavelength

Wavelength of the propagating field.

Corr Length

Physical extent of nonuniformity correlation.

Fluctuation

Magnitude of nonuniformity fluctuation.

dn/dx

Index gradient in x direction.

dn/dz

Index gradient in z direction.

dloss/dx

Grating out coupling coefficient.

dloss/dz

Grating wavenumber.

Layer#

User input layer number, the following parameters (Index, Loss, Thickness, Doping) will display the layer information corresponding to this layer number.

Layer Name

Name of selected layer.

Index

Index of refraction of selected layer.

Loss

Loss of selected layer.

Thickness

Thickness of selected layer.

Doping

Doping of selected layer.

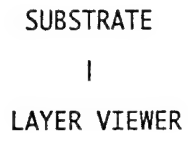
OUTPUTS

Layer data (laydata user data type)

The layer data output contains the substrate parameters and the parameters for each layer in the structure, identical to the input layer data.

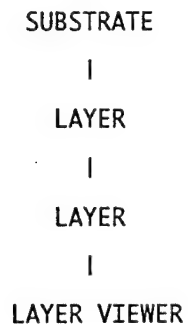
EXAMPLE 1

This example displays a layer viewer module being used to view the substrate parameters:



EXAMPLE 2

This example displays a layer viewer module being used to view the substrate parameters and the layer information in a structure:



RELATED MODULES

substrate
layer

NAME

modeig - calculate effective index for a composite material

SUMMARY

Name modeig

Availability OEIC module library

Type material

Inputs laydata (layer user data type)

Outputs matdata (material user data type)

Parameters

Name	Type	Default
Qzr	float	11.7
Qzi	float	0.0
Input	boolean	false
Layer	boolean	false
Dbase	boolean	false
Output	boolean	false
Nfield	boolean	false
Ffield	boolean	false
Phm	boolean	false
Gamma	boolean	false
Wzr	boolean	false
Wzi	boolean	false
Qzr	boolean	false
Qzi	boolean	false
Qzmr	boolean	false
Qzmi	boolean	false
Km	boolean	false
It	boolean	false
Fwhpn	boolean	false

Fwhpf	boolean	false
Ra	boolean	false
GamLay	string	Null
CompGam	boolean	false

DESCRIPTION

The modeig module calculates the effective index and loss for the input material composite. The module outputs these values as part of the matdata output. Other calculated parameters are available as file output by toggling the output buttons.

INPUTS

Layer data (laydata user data type)

The layer input usually comes from a substrate or layer module and is the data describing the material composite.

PARAMETERS

Qzr

Qzi

Input

Toggles the input file write flag.

Layer

Toggles the layer file write flag.

Dbase

Toggles the dbase file write flag.

Output

Toggles the output file write flag.

Nfield

Toggles the near field file write flag.

Ffield

Toggles the far field file write flag.

Phm

Toggles the dbase file phm write flag.

Gamma

Toggles the dbase file Gamma write flag.

Wzr

Toggles the dbase file Wzr write flag.

Wzi

Toggles the dbase file Wzi write flag.

Qzr

Toggles the dbase file Qzr write flag.

Qzi

Toggles the dbase file Qzi write flag.

Qzmr

Toggles the dbase file Qzmr write flag.

Qzmi

Toggles the dbase file Qzmi write flag.

Km

Toggles the dbase file Km write flag.

It

Toggles the dbase file It write flag.

Fwhpn

Toggles the dbase file Fwhpn write flag.

Fwhpf

Toggles the dbase file Fwhpf write flag.

Ra

Toggles the dbase file ra write flag.

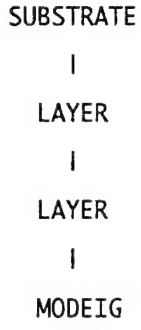
OUTPUTS

Material data (matdata user data type)

The matdata output contains the material parameters along with the calculated effective index and loss.

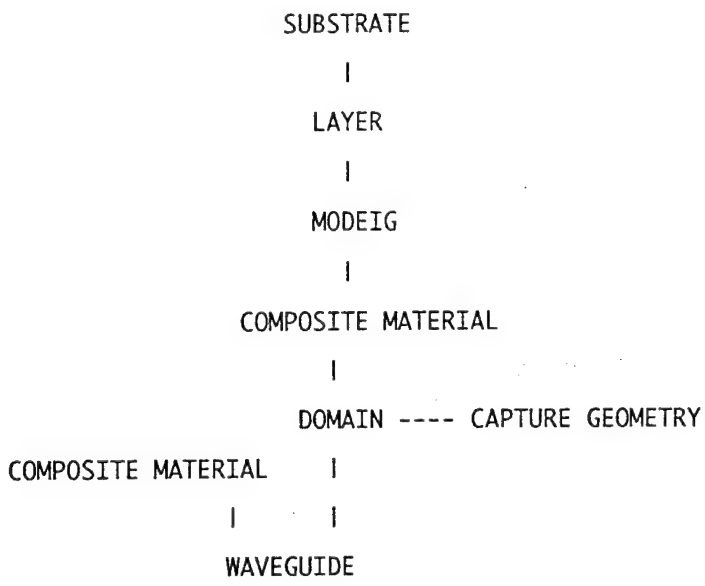
EXAMPLE 1

This example displays a modeig module being used to calculate the effective index and loss of a material composite:



EXAMPLE 2

This example displays a modeig module being used to calculate the effective index and loss of a material composite and then passing the data on for use in a geometry:



RELATED MODULES

substrate
layer
composite material

SEE ALSO

Modeig/II users manual.

NAME

normalize source - normalizes a source to its area

SUMMARY

Name	normalize source
Availability	OEIC module library
Type	processing
Inputs	srcdata (source user data type)
Outputs	field 2-D 1-vector rectilinear 2-space float srcdata (source user data type)
Parameters	NONE

DESCRIPTION

The normalize source module normalizes a source mode to its area.

INPUTS

Source data (srcdata user data type)

The source input usually comes from the source module.

PARAMETERS

NONE

OUTPUTS

field 2-D 1-vector rectilinear 2-space float

The output field which contains the normalized source mode data.

Source data (srcdata user data type)

The srcdata output contains the source information (wavelength, power, mode shift, filename) and the normalized solution mode data (normalized to its area).

EXAMPLE 1

This example displays a normalize source module being used to normalize a source modules output:

SOURCE

|

NORMALIZE SOURCE

|

BPM

RELATED MODULES

source
source viewer
regrid source
overlap
bpm

Release 1 OEIC normalize source()

NAME

object viewer - view the geometry object data port parameters

SUMMARY

Name object viewer

Availability OEIC module library

Type geometry

Inputs objectdata (user data type)

Outputs objectdata (user data type)

Parameters

Name	Type
Id	string
Type	string
Width	string
Length	string
Global X	string
Z	string
Skew	string
Flare	string
Eff Index	string
Eff Loss	string
Curr Density	string

DESCRIPTION

The object viewer module allows the user to view the geometry object parameters and axis data coming from an objectdata (user data type) output port.

INPUTS

Object data (objectdata user data type)

The object data input usually comes from a geometry module (domain, waveguide, ...) and contains the geometry object parameters and the axis specification for the next geometry module.

PARAMETERS

Id

Displays the module id of the geometry.

Type

Displays the geometry type.

Width

Displays the domain width of the geometry.

Length

Displays the domain length of the geometry.

GLobal X

X grid starting position for next geometry.

Z

Z grid starting position for next geometry.

Skew

Skew angle of geometry.

Flare

Flare angle of geometry.

Eff Index

Index of refraction.

Eff Loss

Loss.

Curr Density

Current Density.

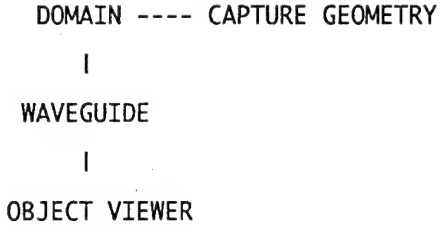
OUTPUTS

Object data (objectdata user data type)

The object data input usually comes from a geometry module (domain, waveguide, ...) and contains the geometry object parameters and the axis specification for the next geometry module.

EXAMPLE 1

This example displays an object viewer module being used to view the waveguide object parameters and its axis information:



RELATED MODULES

domain

capture geometry

Release 1 OEIC object viewer()

NAME

overlap - normalizes a source to its area

SUMMARY

Name	overlap	
Availability	OEIC module library	
Type	processing	
Inputs	srcdata (source user data type)	
	srcdata (source user data type)	
Outputs	float	
Parameters	Overlap	float

DESCRIPTION

This procedure computes the overlap integral between two sources. The overlap is the amount of power transferred from the second source to first. The first source is the reference and it must be normalized to unity area. The second source contains the power.

INPUTS

Source data (srcdata user data type)

This source input usually comes from the source module and must be normalized to unity area.

Source data (srcdata user data type)

This source input usually comes from the source module and must contain the power.

PARAMETERS

Overlap

The overlap integral.

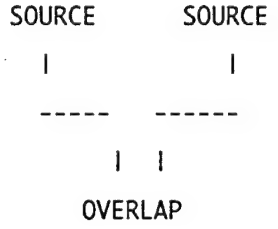
OUTPUTS

float

The overlap integral.

EXAMPLE

This example displays an overlap module being used to compute the overlap integral of two sources:



RELATED MODULES

source
source viewer
normalize source
regrid source

NAME

regrid source - regrids a source mode onto the input geometric grid

SUMMARY

Name regrid source

Availability OEIC module library

Type processing

Inputs srcdata (source user data type)
geomdata (geometry user data type)

Outputs srcdata (source user data type)
field 2D rectilinear float

Parameters

Name	Type	Default	Units
Shift	float	0.0	um

DESCRIPTION

This procedure regrids a source mode onto the input geometric grid.

INPUTS

Source data (srcdata user data type)

This source input usually comes from the source module.

Geometry data (geomdata user data type)

This geometry input usually comes from the capture geometry module.

PARAMETERS

Shift

Defines the x-directional shift of the mode from center (microns).

OUTPUTS

field 2-D 1-vector rectilinear 2-space float

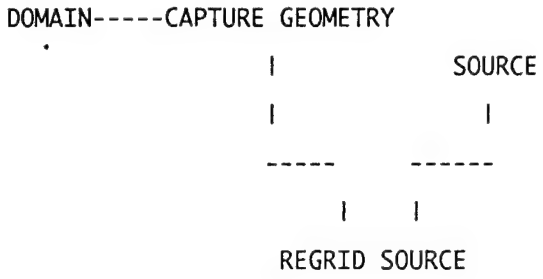
The output field which contains the normalized source mode data.

Source data (srcdata user data type)

The srcdata output contains the source information (wavelength, power, mode shift, filename) and the normalized solution mode data (normalized to its area).

EXAMPLE

This example displays a regrid source module being used to regrid an existing source mode to a different geometry grid:



RELATED MODULES

source
source viewer
normalize source
capture geometry

NAME

region viewer - view the geometry geomdata port parameters

SUMMARY

Name region viewer

Availability OEIC module library

Type geometry

Inputs geomdata (user data type)

Outputs geomdata (user data type)

Parameters

Name	Type
Width	string
Length	string
Nx	string
Nz	string
Zstep	string
Alpha	string
Wavelength	string
Neff	string
Loss	string
Corr Length	string
Fluctuation	string
dn/dx	string
dn/dz	string
dloss/dx	string
dloss/dz	string
Region#	typein-integer
Type	string
Eff Index	string
Eff Loss	string

Curr Density	string
Width	string
Length	string
Skew	string
Flare	string
Z	string
X	string

DESCRIPTION

The region viewer module allows the user to view the geometry object parameters and axis data coming from an geomdata (user data type) output port. By typing in the desired region number the corresponding object information will be displayed.

INPUTS

Geometry data (geomdata user data type)

The geometry data input usually comes from the capture geometry module and contains the material substrate and indexes, and the geometry domain, object, and axis parameters.

PARAMETERS

Width

Displays the domain width of the geometry.

Length

Displays the domain length of the geometry.

Nx

Number of grid points in the transverse dimension.

Nz

Number of grid points in the longitudinal dimension.

Zstep

Length of propagation step for the numerical solution.

Alpha

Sets a nonuniform x grid. If Alpha < 1, the x grid is denser at the center; if Alpha > 1, the x grid is denser at the edges.

Wavelength

Wavelength of the propagating field.

Neff

Index of refraction.

Loss

Loss.

Corr Length

Physical extent of nonuniformity correlation.

Fluctuation

Magnitude of nonuniformity fluctuation.

dn/dx

Index gradient in x direction.

dn/dz

Index gradient in z direction.

dloss/dx

Grating out coupling coefficient.

dloss/dz

Grating wavenumber.

Region #

Select the region of interest and display its geometry parameters.

Eff Index

Index of refraction.

Eff Loss

Loss.

Curr Density

Current Density.

Width

Displays the width of the selected region.

Length

Displays the length of the selected region.

Skew

Skew angle of geometry.

Flare

Flare angle of geometry.

Z

Z grid starting position for next geometry.

X

X grid starting position for next geometry.

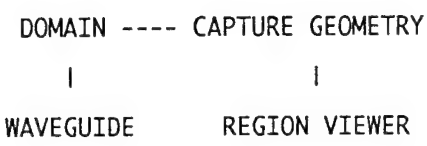
OUTPUTS

Geometry data (geomdata user data type)

The geometry data output contains the material substrate and indexes, and the geometry domain, object, and axis parameters, it is identical to the geometry input port.

EXAMPLE 1

This example displays a region viewer module being used to view the geometry parameters:



RELATED MODULES

capture geometry
domain
substrate
composite material

Release 1 OEIC region viewer()

NAME

source - generate or read the mode data from a file

SUMMARY

Name source

Availability OEIC module library

Type processing

Inputs geomdata (geometry user data type) [OPTIONAL]

Outputs field 2-D 1-vector rectilinear 2-space float
srcdata (source user data type)

Parameters

Name	Type	Default
Method	choice	Numerical
Power	float	0.0
Width	float	15.0 or geometry domain width
Mode	integer	1
Mode Shift	float	0.0
Filename	browser	""
Go	oneshot	0

DESCRIPTION

The source module calculates the solution modes corresponding to the input geometry or reads the modes from a '.mod' file. The module outputs the selected mode as a field and as part of the srcdata user data type. The srcdata output port contains the mode data normalized to a peak value of 1 and the initial power, wavelength, width, mode shift, and # of points. The field output port contains the raw mode data and the x-coordinate data. The geometry input port must be connected to generate new mode data but is not necessary when using the file browser. This module is used mainly as the source input for the bpm module.

INPUTS

Geometry data (optional; geomdata user data type)

The geometry input usually comes from the capture geometry module and is a geometry specification using the geomdata user data type. The geometry input port must be connected to generate new mode data but is not necessary when using the file browser.

PARAMETERS

Method

Selects the method of generating the solution modes. A set of choices are displayed allowing the user to select Numerical, Gaussian, or File Browser.

Power

Defines the initial power of the solution mode.

Width

Defines the enclosing width over which to calculate the solution mode (microns).

Mode

Selects which order mode to output.

Mode Shift

Defines the x-directional shift of the mode from center (microns).

Filename

Defines the filename to read/write the solution modes from/to.

Go

Starts the calculation of the solution modes. This button is not visible when reading modes using the File Browser.

OUTPUTS

field 2-D 1-vector rectilinear 2-space float

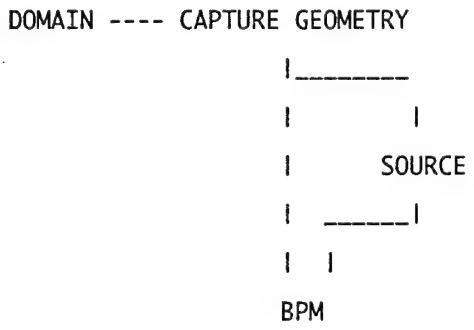
The output field contains the raw source mode data.

Source data (srcdata user data type)

The srcdata output contains the source information (wavelength, power, mode shift, filename) and the normalized solution mode data (normalized to peak value of 1).

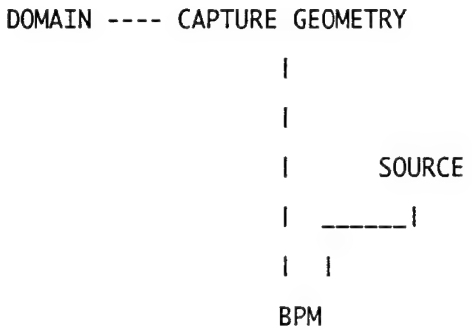
EXAMPLE 1

This example displays a source being generated from a geometry and input to the bpm module:



EXAMPLE 2

This example displays a source being read from a file, using the File Browser, and input to the bpm module:



RELATED MODULES

source viewer

bpm

capture geometry

SEE ALSO

The demos FOAL, COUPLER, and WAVE, as well as others demonstrate the source module.

NAME

source viewer - view the source srcdata ports parameters and data

SUMMARY

Name source viewer

Availability OEIC module library

Type processing

Inputs srcdata (source user data type)

Outputs field 2-D 1-vector rectilinear 2-space float
srcdata (source user data type)

Parameters

Name	Type
Filename	string
Wavelength	string
Power	string
Mode Shift	string
Mode	string
Nx	string
Width	string

DESCRIPTION

The source viewer module allows the user to view the parameters and data coming from a srcdata (user data type) output port. The source filename, wavelength, power, mode shift, # of points, and mode selected are displayed. The normalized mode data is available on the field output port and the source output port is identical to the source input port.

INPUTS

Source data (srcdata user data type)

The srcdata input usually comes from the source module and contains the normalized mode data and the parameters used to generate the mode.

PARAMETERS

Filename

Displays the filename to read/write the solution modes from/to.

Wavelength

Displays the wavelength of the solution mode.

Power

Displays the initial power of the solution mode.

Mode Shift

Displays the x-directional shift of the mode from center (um).

Mode

Displays the selected mode number.

Width

Displays the domain width over which the mode was calculated.

OUTPUTS

field 2-D 1-vector rectilinear 2-space float

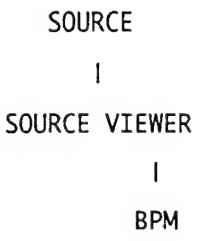
The output field contains the normalized source mode data.

Source data (srcdata user data type)

The srcdata output contains the source information (wavelength, power, mode shift, filename) and the normalized mode data.

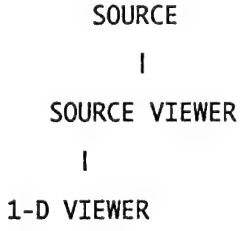
EXAMPLE 1

This example displays the source viewer being used to view the source parameters being passed from the source module to the bpm module:



EXAMPLE 2

This example displays the source viewer being used to view the raw source mode data (not normalized) generated by the source module:



RELATED MODULES

source
bpm

NAME

substrate - generate the material substrate parameters

SUMMARY

Name substrate

Availability OEIC module library

Type material

Inputs none

Outputs laydata (layer user data type)

Parameters

Name	Type	Default	Units
Layer Name	string	substrate	
Wavelength	float	0.9	um
Material	choice	GaAlAs	
Aluminum	float	0.0	%
Q	float	0.0	
Index	float	0.0	
Loss	float	0.0	1/um
Thickness	float	0.0	um
Doping	float	0.0	1/um3
Random	boolean		
Linear	boolean		
Corr Length	float	0.0	um
Fluctuation	float	0.0	
dn/dx	float	0.0	1/um
dn/dz	float	0.0	1/um
dloss/dx	float	0.0	1/um2
dloss/dz	float	0.0	1/um2

DESCRIPTION

The substrate module allows the user to enter the material substrate parameters.

PARAMETERS

Layer Name

Identifying name of the substrate layer.

Wavelength

Wavelength of the propagating field (microns).

Material

Material composition of the substrate {GaAlAs, InGaAsP, Other}.

Aluminum

Percentage of aluminum in GaAlAs material used to calculate the index of refraction. This parameter will only be visible when GaAlAs is selected as the layer material. (Range = 0-100%)

Q

Q value of InGaAsP material used to calculate the index of refraction. This parameter will only be visible when InGaAsP is selected as the layer material. (Range = 0-1.55)

Index

Index of refraction. This parameter will only be visible when Other is selected as the layer material.

Loss

Loss (inverse microns).

Thickness

Thickness of substrate layer (microns).

Doping

Doping of substrate (inverse cubed microns).

Random Nonuniformity

Displays random nonuniformity parameters.

Linear Nonuniformity

Displays linear nonuniformity parameters.

Corr Length

Physical extent of random nonuniformity correlation (microns).

Fluctuation

Magnitude of random nonuniformity fluctuation.

dn/dx

Index gradient in x direction for linear nonuniformity.

dn/dz

Index gradient in z direction for linear nonuniformity.

dloss/dx

Grating out coupling coefficient for linear nonuniformity.

dloss/dz

Grating wavenumber for linear nonuniformity.

OUTPUTS

Layer data (laydata user data type)

The layer data output contains the substrate parameters.

EXAMPLE 1

This example displays the substrate module being used in a composite material:

SUBSTRATE

|

LAYER

|

MODEIG

RELATED MODULES

layer viewer

layer

modeig

composite material

Release 1 OEIC substrate()

NAME

transmission - normalizes a source to its area

SUMMARY

Name	transmission
Availability	OEIC module library
Type	processing
Inputs	srcdata (source user data type) srcdata (source user data type)
Outputs	float
Parameters	transmission float

DESCRIPTION

This procedure computes the transmission integral between two sources. The transmission is the amount of power transferred from the second source to first. The first source is the reference and it must be normalized to unity area. The second source contains the power.

INPUTS

Source data (srcdata user data type)

This source input usually comes from the source module and must be normalized to unity area.

Source data (srcdata user data type)

This source input usually comes from the source module and must contain the power.

PARAMETERS

transmission

The transmission integral.

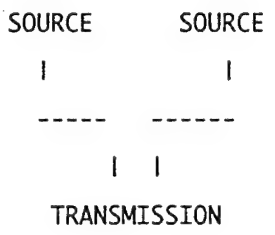
OUTPUTS

float

The transmission integral.

EXAMPLE

This example displays an transmission module being used to compu
transmission
integral of two sources:



RELATED MODULES

overlap
source
source viewer
normalize source
regrid source

NAME

trigger - takes any type input and outputs a trigger pulse when any input changes

SUMMARY

Name	trigger	
Availability	OEIC module library	
Type	processing	
Inputs	integer	
	float	
	string	
	field	
Outputs	oneshot	
	boolean	
	integer	
Parameters	Invert	boolean

DESCRIPTION

The trigger module takes any type input and outputs a trigger pulse when any input changes. There are 3 different outputs for the users convenience, all are pulsed simultaneously.

INPUTS

integer

The integer input.

float

The float input.

string

The string input.

field

The field input.

PARAMETERS

Invert

Inverts the trigger pulse output, instead of outputting a one when the input changes and zero otherwise, the output will be a zero when the input changes and one otherwise.

OUTPUTS

oneshot

The oneshot output, will be pulsed when the input changes.

boolean

The boolean output, will be pulsed when the input changes.

integer

The integer output, will be pulsed when the input changes.

EXAMPLE

This example displays a trigger module being used to start the bpm module when the loop value changes. The bpm module has a oneshot 'Go' button which activates it, making this parameter a visible port and connecting it to the oneshot output of the trigger module allows the bpm module to run in a loop.

RELATED MODULES

float loop

Release	1	OEIC	trigger()

NAME

waveguide - create a geometry waveguide defined by the entered parameters

SUMMARY

Name waveguide

Availability OEIC module library

Type geometry

Inputs

axisData	(geometry axis user data type)	
compdata	(composite material user data type)	

Outputs

objectData	(geometry object user data type)	[INVISIBLE]
axisData	(geometry axis user data type)	

Parameters

Name	Type	Default	Units
Title	string_block	---	---
Change Width	float	0.0	um
Length	float	100.0	um
Flare	float	0.0	degrees
Skew	float	0.0	degrees
Offset X-axis	float	0.0	um
Offset Z-axis	float	0.0	um
Curr. Density	float	0.0	A/um2

DESCRIPTION

The waveguide module creates a geometry waveguide defined by the entered parameters.

Note: The capture geometry module must be present when this module is instantiated so the objectData invisible connection can be made.

INPUTS

axisData (geometry axis user data type)

The geometry axis data, usually comes from a capture geometry module.

compdata (composite material user data type)

The composite material input, usually comes from a composite material module.

PARAMETERS

Title

Displays pertinent information about the waveguide shift geometry and material parameters.

Change Width

Change in the width of the waveguide from the preceding waveguide (microns).

Length

Length of the waveguide (microns).

Flare

Flare half-angle of the waveguide (degrees).

Skew

Skew half-angle of the waveguide (degrees).

Offset X-axis

X-directional offset of the waveguide from the preceding waveguide (microns).

Offset Z-axis

Z-directional offset of the waveguide from the preceding waveguide (microns).

Curr. Density

Current density (pumping) of the waveguide (A/um2).

OUTPUTS

objectData (geometry object user data type) [INVISIBLE]

The geometry object information passed directly to the capture geometry module through an invisible connection.

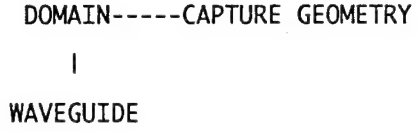
Note: The capture geometry module must be present when this module is instantiated so this invisible connection can be made.

axisData (geometry axis user data type)

The geometry axis information which passes the current axis data to the succeeding geometry module.

EXAMPLE

This example displays a waveguide module being used in a geometry:



RELATED MODULES

domain
capture geometry
curved waveguide
waveguide shift
axis viewer
object viewer

NAME

waveguide shift - create a geometry that shifts the waveguide in the x-direction

SUMMARY

Name waveguide shift

Availability OEIC module library

Type geometry

Inputs

axisData	(geometry axis user data type)
compdata	(composite material user data type)

Outputs

objectData	(geometry object user data type)	[INVISIBLE]
axisData	(geometry axis user data type)	

Parameters

Name	Type	Default	Units
Title	string_block	---	---
Change Width	float	0.0	um
Length	float	100.0	um
X-shift	float	0.0	um
Offset X-axis	float	0.0	um
Offset Z-axis	float	0.0	um
Curr. Density	float	0.0	A/um2

DESCRIPTION

The waveguide shift module create a geometry that shifts the waveguide in the x-direction.

Note: The capture geometry module must be present when this module is instantiated so the objectData invisible connection can be made.

INPUTS

axisData (geometry axis user data type)

The geometry axis data, usually comes from a capture geometry module.

compdata (composite material user data type)

The composite material input, usually comes from a composite material module.

PARAMETERS

Title

Displays pertinent information about the waveguide shift geometry and material parameters.

Change Width

Change in the width of the waveguide from the preceding waveguide (microns).

Length

Length of the waveguide (microns).

X-shift

Total shift of the waveguide in the x-direction (microns).

Offset X-axis

X-directional offset of the waveguide from the preceding waveguide (microns).

Offset Z-axis

Z-directional offset of the waveguide from the preceding waveguide (microns).

Curr. Density

Current density (pumping) of the waveguide (A/um2).

OUTPUTS

objectData (geometry object user data type) [INVISIBLE]

The geometry object information passed directly to the capture geometry module through an invisible connection.

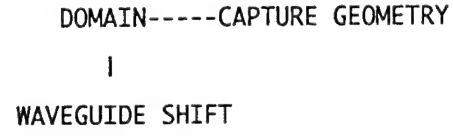
Note: The capture geometry module must be present when this module is instantiated so this invisible connection can be made.

axisData (geometry axis user data type)

The geometry axis information which passes the current axis data to the succeeding geometry module.

EXAMPLE

This example displays a waveguide shift module being used in a geometry:



RELATED MODULES

domain
capture geometry
waveguide
curved waveguide
axis viewer
object viewer

NAME

y-join - create a y-join geometry defined by the entered parameters

SUMMARY

Name y-join

Availability OEIC module library

Type geometry

Inputs

axisData	(geometry axis user data type)
compdata	(composite material user data type)

Outputs

objectData	(geometry object user data type) [INVISIBLE]
axisData	(geometry axis user data type)

Parameters

Name	Type	Default	Units
Title	string_block	---	---
Change Width	float	0.0	um
Length	float	100.0	um
Flare	float	0.5	degrees
Skew	float	0.0	degrees
Offset X-axis	float	0.0	um
Offset Z-axis	float	0.0	um
Curr. Density	float	0.0	A/um2

DESCRIPTION

The y-join module creates a geometry waveguide defined by the entered parameters. It is used to join to separate waveguides together into one waveguide. The degree of split of the initial two waveguides is defined by the flare half-angle parameter.

Note: The capture geometry module must be present when this module is instantiated so the objectData invisible connection can be made.

INPUTS

axisData (geometry axis user data type)

The geometry axis data, usually comes from a capture geometry module.

compdata (composite material user data type)

The composite material input, usually comes from a composite material module.

PARAMETERS

Title

Displays pertinent information about the waveguide shift geometry and material parameters.

Change Width

Change in the width of the waveguide from the preceding waveguide (microns).

Length

Length of the waveguide (microns).

Flare

Defines the half-angle degree of split of the two initial waveguides (degrees).

Skew

Skew half-angle of the waveguide (degrees).

Offset X-axis

X-directional offset of the waveguide from the preceding waveguide (microns).

Offset Z-axis

Z-directional offset of the waveguide from the preceding waveguide (microns).

Curr. Density

Current density (pumping) of the waveguide (A/um2).

OUTPUTS

objectData (geometry object user data type) [INVISIBLE]

The geometry object information passed directly to the capture geometry module through an invisible connection.

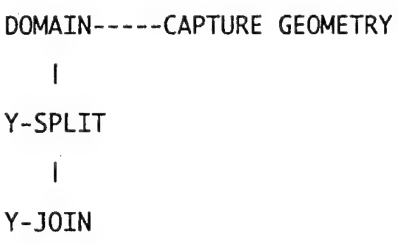
Note: The capture geometry module must be present when this module is instantiated so this invisible connection can be made.

axisData (geometry axis user data type)

The geometry axis information which passes the current axis data to the succeeding geometry module.

EXAMPLE

This example displays a y-join module being used in a geometry:



RELATED MODULES

domain

capture geometry

waveguide

y-split

axis viewer

object viewer

Release	1	OEIC	y-join()

NAME

y-split - create a y-split geometry defined by the entered parameters

SUMMARY

Name y-split

Availability OEIC module library

Type geometry

Inputs

axisData	(geometry axis user data type)
compdata	(composite material user data type)

Outputs

objectData	(geometry object user data type)	[INVISIBLE]
axisData	(geometry axis user data type)	

Parameters

Name	Type	Default	Units
Title	string_block	---	---
Change Width	float	0.0	um
Length	float	100.0	um
Flare	float	0.5	degrees
Skew	float	0.0	degrees
Offset X-axis	float	0.0	um
Offset Z-axis	float	0.0	um
Curr. Density	float	0.0	A/um2

DESCRIPTION

The y-split module creates a geometry waveguide defined by the entered parameters. It is used to split a single waveguide into two separate waveguides. The degree of split of the resulting two waveguides is defined by the flare half-angle parameter.

Note: The capture geometry module must be present when this module is instantiated so the objectData invisible connection can be made.

INPUTS

axisData (geometry axis user data type)

The geometry axis data, usually comes from a capture geometry module.

compdata (composite material user data type)

The composite material input, usually comes from a composite material module.

PARAMETERS

Title

Displays pertinent information about the waveguide shift geometry and material parameters.

Change Width

Change in the width of the waveguide from the preceding waveguide (microns).

Length

Length of the waveguide (microns).

Flare

Defines the half-angle degree of split of the two waveguides (degrees).

Skew

Skew half-angle of the waveguide (degrees).

Offset X-axis

X-directional offset of the waveguide from the preceding waveguide (microns).

Offset Z-axis

Z-directional offset of the waveguide from the preceding waveguide (microns).

Curr. Density

Current density (pumping) of the waveguide (A/um2).

OUTPUTS

objectData (geometry object user data type) [INVISIBLE]

The geometry object information passed directly to the capture geometry module through an invisible connection.

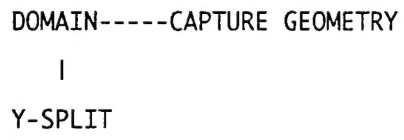
Note: The capture geometry module must be present when this module is instantiated so this invisible connection can be made.

axisData (geometry axis user data type)

The geometry axis information which passes the current axis data to the succeeding geometry module.

EXAMPLE

This example displays a y-split module being used in a geometry:



RELATED MODULES

domain

capture geometry

waveguide

y-join

axis viewer

object viewer

Index